Pzpu #32 LOC: 47

OIPE JC64 OCT 24 2003 PATENT & TRADEMARK OFFICE

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

TC 2100, CPK2, 2911

Patent No: 5,838,906

Issued: November 17, 1998

For: Distributed Hypermedia Method for Automatically Invoking External Application Providing Interaction and Display of Embedded Objects within a Hypermedia Document

RECEIVED
JAN 2 9 2004
Technology Center 2100

CITATION OF PRIOR ART UNDER 35 U.S.C. § 301 AND 37 CFR 1.501 IN RELATION TO U.S. PATENT NO. 5,838,906

RECEIVED
OCT 2 9 2003
OFFICE OF PETITIONS

Mail Stop: Prior Art Department (Citation of Prior Art per 37 CFR 1.501)
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Dear Sir,

On behalf of the World Wide Web Consortium, the primary standard-setting organization for the World Wide Web,[1] please find enclosed two prior art publications to be included in the file wrapper of U.S. Patent No. 5,838,906 ("the '906 patent") pursuant to 35 U.S.C. § 301 and 37 C.F.R. §1.501. The enclosed publications are prior art to the '906 patent under 35 U.S.C. § 102(b). They were never considered by the United States Patent & Trademark Office during the prosecution of the '906 patent. These publications, taken alone, anticipate at least claims 1, 2, 3, 6, 7 and 8 of the '906 patent, and, taken together with the Mosaic browser that was acknowledged in the patent as prior art, plainly render those claims invalid as obvious under 35 U.S.C. § 103.

As the Commissioner may be aware, the '906 patent is the subject of a patent infringement suit brought by Eolas Technologies, Inc. and the Regents of the University of California (the patent's exclusive licensee and owner, respectively) against Microsoft Corporation. The suit alleged that Microsoft's Internet Explorer, the most widely used program in the world for browsing the World Wide Web, infringed claims of the '906 patent. A jury in that case recently found against Microsoft and awarded Eolas and the University of California in excess of $500 million. Microsoft is appealing that verdict, but has also stated publicly that it intends in any event to redesign Internet Explorer in a manner that it believes plainly does not infringe the '906 patent. Although Microsoft's proposed redesign, as we understand it, involves only a small portion of

[1] The World Wide Web is a network of information resources that can be accessed through the Internet. A list of the member companies of the World Wide Web Consortium is available at http://www.w3.org/Consortium/Member/List.

Internet Explorer, it would render Microsoft's browser incompatible with globally-accepted standards and impair the operation of millions of Web pages. The cost to the larger World Wide Web community of fixing the problems created by such a change to Internet Explorer is incalculable, but would likely require changes to millions of Web pages, as well as changes to Web page authoring tools and other software and systems designed for the World Wide Web. This enormous expense and attendant incalculable disruption, not to mention the threat the '906 patent as construed by the court poses to other browsers widely used in the Web community, are completely unwarranted because we strongly believe that the '906 patent is invalid in view of prior art, submitted herewith, that was never previously considered by the United States Patent & Trademark Office. While we understand that the submitted prior art was introduced during the course of the recent trial proceedings, the issue of whether it renders the '906 patent invalid was never considered.[2] In view of the pervasive negative impact of the '906 patent on the larger World Wide Web community, which is unwarranted in view of the patent's invalidity, the World Wide Web Consortium believes that the Director should, on his initiative, commence a reexamination of the '906 patent.

The '906 patent is generally directed to a Web browser able to invoke external programs to display portions of a Web page that the browser cannot directly display itself. A Web browser may not be capable of displaying certain types of image data, for example, in which case the browser would invoke a separate program that is capable of doing so. The sole difference between the web browser described in the '906 patent and typical browsers that the patent acknowledges as prior art, is that with prior art browsers, the image in such cases is displayed in its own window, separate from the main browser window, whereas, with the '906 browser the image is displayed in the same window as the rest of the Web page, without the need for a separate window. But that feature (i.e., displaying, or embedding, an image generated by an external program in the same window as the rest of a Web page) had already been described in the prior art publications submitted herewith and was known to the Web development community. The claims of the '906 patent are therefore plainly obvious in view of this prior art.

Even prior to the development of this feature in Web browsers, software developers had recognized the usefulness of adding the same functionality to prior art word processing programs, which display documents instead of Web pages. For example, more than a year before the '906 patent was filed, a word processing program called Write, provided with Microsoft Windows 3.1, enabled users to embed into Write documents graphic images created with the Paint program. The Write program would invoke the Paint program to display the illustration within the same window as the rest of the document. The '906 patent thus added nothing to the art — it only applied a well known concept in the display of documents to the display of Web pages, and even then, did so after the enclosed Raggett publications had disclosed the same thing for web pages.

The two enclosed references are printed publications published more than one year prior to the filing date of the '906 patent. Each is therefore prior art to the '906 patent under 35 U.S.C. § 102(b). Neither reference was cited, made of record or considered during the prosecution of the '906 patent. One set of copies is provided for inclusion in the file wrapper of the '906 patent. The second set of copies is provided to permit service by the Office on the patent owner.

---

[2] We understand the court entered a judgment as a matter of law that other prior art (but not the two Raggett publications) differed from the claimed subject matter and that the issue of invalidity over the Raggett publications was not put to the jury or otherwise considered.

The Raggett I and Raggett II Publications

The two enclosed publications relate to HTML+, a proposed specification extending the features of Hypertext Markup Language ("HTML"), the standard language in which Web pages were, and still are, written. The first publication ("*Raggett I*," Exhibit A hereto) is a draft of the HTML+ specification, which was made publicly available for comment on July 23, 1993. *Raggett I* was authored by Dave Raggett, a researcher at Hewlett Packard Laboratories, who attempted in that document to pull together comments regarding extensions to HTML from the participants in www-talk, a public mailing list hosted by Tim Berners-Lee, the founder of the Web and now the Director of the World Wide Web Consortium. The second publication ("*Raggett II*," Exhibit B hereto) is a message posted on June 14, 1993 to the public www-talk mailing list, describing the EMBED tag in HTML+. The EMBED tag described in *Raggett I* and *II* is identical in all material respects to the EMBED tag described in the '906 patent, which in turn was the basis for its claims.

As described in *Raggett I*, the EMBED tag enables a browser to display in-line (i.e. without going to a separate browser window) information rendered by an external application or external shared library. That is, it enabled the browser to display the information rendered by the external application, or shared library, in the same window displaying the information rendered by the browser. (*Raggett I*, p. 6, last para.). The example given in *Raggett I* is the display by a browser of an equation rendered using EQN, a program that formats and displays mathematical equations:

<embed type="application/eqn">2 pi int sin (omega t)dt </embed>

Specifically, in this example, "*2 pi int sin (omega t)dt*" is the embedded data to be rendered as a formatted equation and "*type="application/eqn"*" specifies the external program, EQN, capable of rendering that data. *Raggett I* also described using the EMBED tag in combination with the FIG tag in order to display in-line images having data formats that were not recognized by the browser. (*Raggett I*, p. 12).

The particular external program, or shared library, that must be used to render the data in the EMBED tag is identified by the TYPE attribute of the EMBED tag. *Raggett I* used the well-known MIME protocol to identify, locate and invoke an external program or shared library capable of rendering data of the specified type. (*See id.* ("the *type* attribute specifies a registered MIME content type and is used by the browser to identify the appropriate shared library or external filter to use to render the embedded data, e.g., by returning a pixmap")). As is the case with all other HTML tags described in *Raggett I*, the browser performs the related operations for the disclosed EMBED tag automatically upon parsing the tag, without user input. *Raggett I* further disclosed the use of external editor programs that allow users to interact with the displayed object data within the document. (*See id.* ("Sophistocated [sic] browsers can link to external editors for updating and revising embedded data")). The '906 patent discloses a comparable TYPE attribute of an EMBED tag (Table II) and use of the MIME protocol for matching the type information to an external program for displaying foreign data within a Web browser window, precisely as earlier described in *Raggett I*.

Raggett II further explained that the embedded, or "foreign," data that is to be rendered in-line does not need to be contained within the EMBED tag, as in the example in *Raggett I*, but may instead be located in a separate file referenced by a URL. (*See Raggett II*, last sentence). A URL, or Uniform Resource Locator, specifies the location of a file anywhere on the Internet. In addition,

Raggett II repeated the operative description of the EMBED tag operation from *Raggett I* and provided multiple suggestions for implementing the EMBED tag operation. For example, it explained how to bind a MIME type to the appropriate external rendering program ("e.g. via X resources or a config file") and provided suggestions for implementing the external programs (for example, via "separate programs driven via pipes and sdin/stdout or as dynamically linked library modules (Windows DLLs)").

Raggett I also explained that HTML+, including the EMBED tag, is "for use within the World Wide Web" and, in particular, that "[i]nformation browsers can display information . . . in HTML+ format." *Raggett I* at page 1. It further explained that the World Wide Web is a client-server environment in which hypermedia documents are retrieved across the Internet. *Raggett I* at page 1 ("The World Wide Web is a wide area client-server architecture for retrieving hypermedia documents across the Internet.").

Raggett I was widely disseminated in 1993 by and to, among others, the leaders in the effort to standardize the World Wide Web, including the founding participants in the World Wide Web Consortium, again today's leading standard-setting organization for the World Wide Web. The publication was, has been and continues to be available to all interested persons through the Internet and through other means since on or prior to July 23, 1993. As such, it is a "printed publication" within the meaning of 35 U.S.C. §102 (b). *See* M.P.E.P. § 2128 (2003) (stating, in a section entitled "ELECTRONIC PUBLICATIONS AS PRIOR ART: Status as a 'Printed Publication'" that: "An electronic publication, including an on-line database or Internet publication, is considered to be a 'printed publication' within the meaning of 35 U.S.C. 102(a) and (b) provided the publication was accessible to persons concerned with the art to which the document relates."). The effective date of the printed publication is the date of its availability; namely, at least as early as July 23, 1993. *See* M.P.E.P. § 2128 (stating, in section entitled "ELECTRONIC PUBLICATIONS AS PRIOR ART: Date of Availability" that: "Prior art disclosures on the Internet or on an on-line database are considered to be publicly available as of the date the item was publicly posted. If the publication does not include a publication date (or retrieval date), it cannot be relied upon as prior art under 35 U.S.C. 102(a) or (b)."). A dated copy of the document currently can be retrieved from the Cite Seer: Scientific Research Digital Library site via http://citeseer.nj.nec.com/raggett93html.html (a pdf version of *Raggett I*, which can be viewed using Adobe Acrobat, can be retrieved by clicking on the "PDF" hyperlink located in the upper right corner of the Web page). Also, dated entries in the WWW-TALK archives relating to provisions of the HTML+ specification, as well as the original posting of the July 23, 1993 HTML+ specification, are currently available on-line at http://ksi.cpsc.ucalgary.ca/archives/WWW-TALK/www-talk-1993q2.messages/467.html and http://ksi.cpsc.ucalgary.ca/archives/WWW-TALK/www-talk-1993q3.messages/282.html.

Raggett II was also widely disseminated and publicly available through the Internet and through other means at least since June 14, 1993, and is currently available on-line at http://ksi.cpsc.ucalgary.ca/archives/WWW-TALK/www-talk-1993q2.messages/467.html. It is a "printed publication" within the meaning of 35 U.S.C. §102(b) because it was a "contribution" to "electronic bulletin boards, message systems, and discussion lists" that were "accessible to the persons concerned with the art to which the document relates" when it was posted to the WWW-Talk list (see, e.g., M.P.E.P. §§ 707.05(e), 2128).[3] It enjoys prior art effect as of the date of its

[3] For instance, a review of the University of Calgary archive site containing this posting demonstrates that more than 1,000 such postings were made during the three months surrounding the posting of the July 23rd HTML+ Specification (*Raggett I*) by the very people that were

posting (i.e., June 14, 1993), pursuant to M.P.E.P. § 2128 (see, e.g., "ELECTRONIC PUBLICATIONS AS PRIOR ART: Date of Availability").

The NSCA Mosaic Web Browser and Other Acknowledged Prior Art

The '906 patent acknowledges that Web browsers were in the prior art and in fact describes its alleged invention in terms of modifications to one such prior art browser, the NCSA Mosaic browser, Version 2.4. *See, e.g.*, '906 patent, column 3, lines 9 to 12 (stating that "An example of a browser program is the National Center for Supercomputing Application's (NCSA) Mosaic software developed by the University of Illinois at Urbana/Champaign, Ill."); *see also id.*, column 8, lines 9 to 12 ("[t]he source code in Appendix A includes NCSA Mosaic version 2.4 source code along with modifications to the source code to implement the present invention")(emphasis added); *id.*, column 13, lines 43 to 46 (stating "that much of the source code in is [sic] pre-existing NCSA Mosaic code" and that "[o]nly those portions of the source code that relate to the new functionality discussed in this specification should be considered as part of the invention."). The patent thus acknowledges that the features of Web browsers, at least to the degree reflected in version 2.4 of the NCSA Mosaic Web browser, were prior art to the claimed inventions.

NSCA Mosaic Web browser, version 2.4, like all Web browsers, is a computer program that enabled users to retrieve documents over the Internet and display those documents on a computer monitor. Such documents may contain, for example, "an icon, or other indicator, within the text" linked to a particular image file that users "may select ... to obtain the full image." (See '906 patent, column 2, line 64 to 65, column 3, lines 2 to 3). When a user selects such an indicator, the Mosaic program "retrieves the corresponding full image ... and displays it by using external software" "in a separate window." (*Id.*, column 3, lines 5-7, 16-18; *see also* column 2, line 56 through column 3, line 26 (describing the capabilities of the Mosaic browser, among others).

Differences Between the Claimed Invention and the Prior Art

The sole difference between claims 1 and 6[4] and the NCSA Mosaic browser, Version 2.4, is that the claims require a browser to process a so-called "embed text format," and the Mosaic browser did not have this capability as claimed. In particular, the claimed browser must process an "embed text format" that specifies the location of an "object external" to a hypermedia document (*i.e.*, a document of the type typically displayed by browsers, containing text as well as non-text portions such as graphics, video, sound, etc.). The browser in turn utilizes "type information" associated with the external object to identify, locate and automatically invoke an external "application" that enables the browser to display the object within the hypermedia document being displayed in a browser-controlled window. The '906 patent asserts that the "embed text format" is an improvement over the "helper application" technology employed by prior art browsers such as

---

developing the World Wide Web at the time. (See <http://ksi.cpsc.ucalgary.ca/archives/WWW-TALK/www-talk-1993q3.index.html>.) Moreover, the HTML+ Specification itself asks that comments be sent "to the WWW discussion group: www-talk@nxoc01.cern.ch." (*Raggett I* at page 1, footnote 1.)

[4] Note that claims 1 and 6 are nearly identical but for the type of invention (i.e., claim 1 claims a process, whereas claim 6 is directed to a "computer program product for use in...").

the Mosaic program in which the browser launched an external program in a separate window to display data that it cannot process natively. *See, e.g.,* '906 patent, column 3, lines 2 to 20.

The '906 patent describes the "embed text format" functionality in terms of an EMBED tag. *See, in particular,* '906 patent, column 12, line 54 and Column 13, line 31, Table II and descriptive text. The described EMBED tag has an HREF attribute for specifying the location (e.g., a uniform resource locator, or URL) of an object to be displayed and a TYPE attribute for the MIME type of the object data, which the browser uses to identify, locate and launch an associated application to render that data.

In the context of independent claims 1 and 6, the NCSA Mosaic browser, version 2.4, is a "computer program product" (e.g., a Web browser) that is "embodied" in a "computer usable medium" (e.g., installed in a computer or contained on a disk) for use in a "distributed hypermedia environment" having "at least one client workstation and one network server" (e.g., the Internet). The Mosaic program can run on "said client workstation" to "parse[] a first distributed hypermedia document" (e.g., an HTML document) "received over" the Internet to "identify text formats" (e.g., HTML tags and elements) and "respond[] to predetermined text formats to initiate processing specified by said text formats" in the hypermedia document in order "to display" the document in a browser window on "said client workstation." Furthermore, the Mosaic program can locate "an external object" having "type information associated with it utilized by said browser to identify and to locate an executable application external to" said hypermedia document. The Mosaic program can "invoke" said external application (e.g., an "external editor") "to display" the "external object." As implemented in Mosaic version 2.4, that invocation led to the invoked object being displayed in another window, as opposed to within the browser window displaying the hypermedia document as required by the claims, when the user selected a hyperlink to the external object (as opposed to "automatically" as required by the claims).[5]

The only claim limitation not explicitly disclosed, described and implemented in the admittedly prior art Mosaic browser is the "embed text format" feature, in which a browser "automatically invoke[s]" an external application "to display" an external object within the browser window displaying the hypermedia document. That feature, however, is plainly disclosed in *Raggett I* and *Raggett II* — they specifically describe a substantially identical HTML "embed" tag for automatically invoking an external program to render interactive objects in-line in an HTML document. *Raggett II*, in particular, specifically stated that external, or foreign, data (*i.e.*, an external object) can be contained in a separate file referenced, for example by a URL. Moreover, the ability of a Web browser to retrieve and process data from both local and non-local sources is an inherent feature of such browsers. Indeed, one of the first applications of HTML/Web browsers was the rendering in a document displayed in a single window of text and images, where the image data was contained in files separate from those containing the text.

---

[5]*Raggett I*, for example, also disclosed these same features as the Mosaic Version 2.4 browser. In particular, it disclosed an "information browser[]," *i.e.*, a "computer program product," that can be used to display documents in HTML+ format (a successor to the HTML format then widely in use). *Raggett* I at pages 1-2. It also explained the HTML+ is "for use within the World Wide Web" and that the World Wide Web "is a wide area client-server architecture for retrieving hypermedia documents across the Internet. *Id.* at page 1. It also described "pars[ing] hypermedia documents" (*see id.* at page 3), and "utiliz[ing] [a] browser to display" a hypermedia document (*see id.* at page 1). In general, all the basic browser functions of Mosaic Version 2.4 are inherent in *Raggett I* since such functions are required to display HTML-type hypermedia documents.

An element by element comparison of claim 6-8 to the acknowledged and newly cited prior art is provided below in Table I. It shows that each and every element of each of claims 6-8 is present in the Mosaic version 2.4 browser in combination with *Raggett I* and *Raggett II*, and in *Raggett I* and *II* themselves (*i.e.*, even without relying on Mosaic version 2.4). Claims 1-3 are comparable to claims 6-8, respectively, and each and every element of those claims are also present in the acknowledged and newly cited prior art for the same reasons provided in Table I.

Table I

	Acknowledged Prior Art	**Newly Cited Art**
6. A computer program product for use in a system having at least one client workstation and one network server coupled to said network environment, wherein said network environment is a distributed hypermedia environment, the computer program product comprising: *a computer usable medium having computer readable program code physically embodied therein, said computer program product further comprising: computer readable program code for causing said client workstation to execute a browser application*	Mosaic, see '906 patent at column 1, line 19 to column 3, line 51 (describing the Internet, and the use and function of browser programs, and noting that Mosaic is "an example of a browser program").	*Raggett I* at page 1 (explaining that "HTML+ is a simple SGML based format for wide-area hypertext documents, for use within the World Wide Web," that "[t]he World Wide Web is a wide area client-server architecture for retrieving hypermedia documents across the Internet," and that "[i]nformation browsers can display information ... in the HTML+ format")
to parse a first distributed hypermedia document to identify text formats included in said distributed hypermedia document and to respond to predetermined text formats to initiate processes specified by said text formats;	Mosaic, see '906 patent at column 1, line 19 to column 3, line 51 (same).	*Raggett I* at page 3 (discussing "Parsing HTML+ Documents").
computer readable program code for causing said client workstation to utilize said browser to display, on said client workstation, at least a portion of a first hypermedia document received over said network from said server,	Mosaic, see '906 patent at column 1, line 19 to column 3, line 51 (same).	*Raggett I* at page 1 (explaining that "HTML+ is a simple SGML based format for wide-area hypertext documents, for use within the World Wide Web," and that "[t]he World Wide Web is a wide area client-server architecture for retrieving hypermedia documents across the Internet").
wherein the portion of said first hypermedia document is displayed within a first browser-controlled	Mosaic, see '906 patent at column 1, line 19 to column 3, line 51 (same).	*Raggett I* at page 1 (explaining that "[i]nformation browsers can

Table I		
	Acknowledged Prior Art	**Newly Cited Art**
window on said client workstation,		display information … in the HTML+ format")
wherein said first distributed hypermedia document includes an embed text format, located at a first location in said first distributed hypermedia document, that specifies the location of at least a portion of an object external to the first distributed hypermedia document,	Mosaic, see '906 patent at column 1, line 19 to column 3, line 51 (same).	*See Raggett II* at pages 1-2 (providing an example of an EMBED tag (*i.e.*, an embedded text format) and stating that the foreign (*i.e.*, embedded) data can be put "in a separate file referenced by a URL"). *See also Raggett I* at p. 12 (explaining that the image for the "fig" tag, which is used to display, e.g., images and graphics, can be "defined by a link to an external document.")
wherein said object has type information associated with it utilized by said browser to identify and locate an executable application external to the first distributed hypermedia document	Mosaic, see '906 patent at column 3, lines 5 to 6 (the Mosaic program "retrieves the corresponding full image … and displays it by using external software").	*Raggett I* at page 6 (explaining that the "type attribute" to the EMBED tag "specifies a registered MIME content type and is used by the browser to identify the appropriate shared library or external filter to use to render the embedded data, e.g., by returning a pixmap."); *Raggett II* at page 1 (explaining that "[t]he browser identifies the format of the embedded data from the "type" attribute [to the EMBED tag], specified as a MIME content type;" and further explaining that the type information is used to identify, e.g., a "separate program[]" or "dynamically linked library" for rendering the data).
and wherein said embed text format is parsed by said browser to automatically invoke said executable application on said client workstation	Mosaic, see '906 patent at column 1, line 19 to column 3, line 51 (noting that Mosaic is "an example of a browser program" and, as such, parses HTML documents accessed).	*Raggett I* at pages 3 and 6 (discussing "Parsing HTML+ Documents" generally, and "the EMBED tag" specifically, as part of the automatic processing of an HTML+ document by a Web

Table I		
	Acknowledged Prior Art	**Newly Cited Art**
		browser); *Raggett II* at page 1 (explaining that "[t]he browser identifies the format of the embedded data from the "type" attribute, specified as a MIME content type."). As explained above, *Raggett I* and *II* describe using the "type" attribute to the EMBED tag to identify an external application program or shared library capable of rendering the embedded data. The browser then invokes the identified application or shared library, which in turn returns, for example, "a pixmap." *Raggett I*, p. 6; *Raggett II, p. 1.*
in order to display said object	Mosaic, see '906 patent at column 3, lines 5 to 6 (the Mosaic program "retrieves the corresponding full image … and displays it by using external software").	The purpose of the EMBED tag described in *Raggett I* and *Raggett II* is to display in-line information rendered by an external application program or shared library. *See, e.g., Raggett I* at page 6 (explaining that the "appropriate shared library or external filter [*i.e.*, application program]" is used to "render the embedded data, e.g. by returning a bitmap."). *See also, e.g. Raggett II* at page 1 (explaining that "[b]rowsers can then be upgraded to display new formats without changing their code at all").
and enable interactive processing of said object		*Raggett I* at page 6, line 47 ("Sophistocated [sic] browsers can link to external editors for updating and revising embedded data.").
within a display area created at said first location within the portion of said first distributed hypermedia		*Raggett II* at page 1 (explaining in response to emails regarding embedding

Table I		
	Acknowledged Prior Art	**Newly Cited Art**
document being displayed in said first browser-controlled window.		equations and encapsulated Postscript within documents to be displayed on the Web (*e.g.*, HTML documents) that "both of these will be possible with the HTML+ DTD, by using the capability to embed foreign formats inline in the HTML+ source ...") (emphasis added). *See also Raggett I* at pages 6 and 12 (describing the EMBED tag, which is used to embed data having an external format within a Web page); *see also, id.*, at page 34 (explaining, in a section entitled "Notes for Implementers," that "[i]t is generally better to avoid displaying the retrieved document in a new window, unless explicitly requested by the user.").
7. The computer program product of claim 6, wherein said executable application is a controllable application and further compising: *computer readable program code for causing said client workstation to interactively control said controllable application on said client workstation via inter-process communications between said browser and said controllable application.*		*See Raggett I* at page 6 (describing inter-process communication between the browser and an external editor: "[s]ophistocated [sic] browsers can link to external editors for creating or revising embedded data"). Also *Raggett I* and *II* describe having the browser use shared libraries, such as DLLs, for rendering data in external formats. *Raggett I* at page 6, *Raggett II* at page 1. Such shared libraries would necessarily be controlled through inter-process communications with the browser that invoked them since shared libraries are not independently executable

Table I		
	Acknowledged Prior Art	**Newly Cited Art**
		(that is, they cannot execute unless they are invoked by another program, such as the browser here).
8. The computer program product of claim 7, wherein the communications to interactively control said controllable application continue to be exchanged between the controllable application and the browser even after the controllable application program has been launched.		Again *Raggett I* at page 6 explains that "[s]ophistocated [sic] browsers can link to external editors for creating or revising embedded data". Since the browser displays information rendered by the external program, here the editor, the operation of such an external editor plainly requires continuing communication between the browser and the editor. Otherwise a user would not see displayed the changes being made to the embedded data during the process of revising that data.

Raggett I and *II* Anticipate Claims 1-3 and 6-8

As shown in Table I above, *Raggett I* and *II* collectively disclose each and every element of claims 1-3 and 6-8. In addition, *Raggett I* and *II* comprise a single prior art publication because both were posted on or incorporated by reference in the same Website at the same time more than a year before the filing date of the '906 patent. Specifically, all messages sent to the www-talk email list, including *Raggett II* and a message containing a link to *Raggett I* (*see* Exhibit C hereto), were also posted on the http://eies2.njit.edu:80/wmail.html Website (*see* Exhibit D hereto). Thus, as of July 23, 1993, both *Raggett I* (which is dated July 23, 1993) and *Raggett II* (which is dated June 14, 1993) were effectively published on a single Website. Since *Raggett I* and *II* comprise a single publication and disclose each and every element of claims 1-3 and 6-8, they thus anticipate those claims.

Claims 1-3 and 6-8 are also Obvious Over the Mosaic Version 2.4 Browser in View of *Raggett I* and *Raggett II*

In addition to being anticipated by *Raggett I* and *II*, as set forth above, claims 1-3 and 6-8 are also obvious over the acknowledged Mosaic browser in view of *Raggett I* and *II*. *Raggett I* and *II* specifically teach those of ordinary skill in the art to modify a prior art browser, such as the

Mosaic browser, to incorporate the allegedly new features of claims 1-3 and 6-8, rendering those claims obvious.

The Level of Ordinary Skill in the Art

The person of ordinary skill in the relevant art to the claimed invention is a software programmer with at least a bachelor's degree in Computer Science, and five years of programming experience in Internet, Web and browser technology, including specific experience with programming in HTML. However, even assuming a lower level of ordinary skill in the art, the claims of the '906 patent would still have been obvious, given that the enclosed prior art describe precisely what the '906 patent claims as its invention in precisely the same context.

The Prima Facie Obviousness of Claims 1-3 and 6-8

The printed publications provided herewith were not considered by the PTO during the original prosecution of the '906 patent. When considered in view of the acknowledged prior art (e.g., Mosaic Web browser, version 2.4) by a person of ordinary skill in the art, they render the claimed invention defined by claims 1-3 and 6-8 of the patent *prima facie* obvious.

As described above, the only difference between the claimed invention and the prior art Mosaic browser is that the Mosaic browser was not capable of processing an "embed text format" in a hypermedia document to "automatically invoke" an external application "to display" an external object within the browser window displaying the hypermedia document, as claimed. But *Raggett I* and *Raggett II* however specifically disclose implementing this functionality in a Web browser.

Raggett I and *II* thus provided specific motivation and guidance to a person of ordinary skill to modify the acknowledged prior art NCSA Mosaic version 2.4 browser (and other prior art browsers) to arrive at the claimed invention. Indeed, *Raggett I* (the HTML+ specification), which was publicly disseminated more than a year prior to the filing date of the '906 patent, required Web browsers to possess this functionality in order to be compliant with the proposed specification. As such, it is difficult to envision a document that could have provided greater motivation to modify a Web browser to provide the features called for therein. Furthermore, as acknowledged and admitted by the inventors of the '906 patent (*see, e.g.*, column 13, lines 51 to 59 and column 16, lines 51 to 53), the act of modifying the Mosaic prior art browser to implement the features called for by *Raggett I* and *II* was well within the abilities of a person having an ordinary level of skill in the relevant art (e.g., software programming). *Raggett I* and *Raggett II,* considered individually or in combination, in view of the acknowledged prior art, therefore establish a *prima facie* case of obviousness of claims 1-3 and 6-8.

Further comparison of the '906 patent specification to *Raggett I* and *Raggett II* leaves no doubt as to the accuracy of this conclusion. As described above, Table II (column 12, line 54, with descriptive text through column 13, line 31) of the '906 patent shows the preferred embodiment of an EMBED tag with HREF and TYPE attributes, which the browser uses to identify, locate and launch associated external applications. *Raggett I* and *II* use nearly identical language (*see, e.g. Raggett I*, page 6; *Raggett II*, last sentence) to describe the attributes of the EMBED tag. The enclosed publications thus disclose not only the same functionality but precisely the same means of

implementing that functionality in Web browsers (i.e., the same "EMBED" tag is used to initiate the same browser behavior that provided the same result as the claimed subject matter of the '906 patent).

Moreover, the enclosed publications enable, as the '906 patent claims, Web browsers to provide the user with more functionality (e.g., through displaying and/or editing new data formats) without changing the browser code. Compare, '906 patent, column 11, lines 52 to 55, *Raggett I*, page 6, and *Raggett II*, page 1. Again, the enclosed publications were promulgated to the World Wide Web community more than a year before the filing of the '906 patent for the purpose of implementing this very same capability in prior art Web browsers.

Thus, the two printed publications provided herewith, taken in view of the admittedly prior art NCSA Mosaic version 2.4 browser, provided specific motivation and guidance to persons of ordinary skill to modify the NCSA Mosaic version 2.4 browser to arrive at the claimed invention. As such, these disclosures support a *prima facie* finding of obviousness of claims 1-3 and 6-8 of the '906 patent and render those claims obvious to a person of skill in the art.

Conclusion

The two *Raggett* publications provided herewith anticipate at least claims 1-3 and 6-8 of the '906 patent. In addition, the acknowledged prior art Mosaic version 2.4 browser, when considered together with the two *Raggett* publications, render at least claims 1-3 and 6-8 obvious. In view of the invalidity of these claims and the considerable adverse impact the '906 patent will have on the larger World Wide Web community, a Director initiated reexamination is appropriate.

Respectfully submitted,

Attorneys for Submitter
World Wide Web Consortium.

Date: October 23, 2003

Barry D. Rein (Reg. No. 22,411)
Kenneth L. Stein (Reg. No. 38,704)
PENNIE & EDMONDS LLP
1155 Avenue of the Americas
New York, New York 10036-2711
(212) 790-9090

Exhibit A

Provided by Synapse Group

OIPE JC64 OCT 24 2003 PATENT & TRADEMARK OFFICE

Request for Comments: XXXX

David Raggett (Hewlett Packard)

23rd July 1993

HTML+ (Hypertext markup language)

A proposed standard for a light weight presentation independent delivery format for browsing and querying information across the Internet

Status of this Memo

This document is a proposal for an Internet Draft, and specifies the HTML+ wide-area hypertext document format, with a view to requesting discussion[1] and suggestions for improvements. Distribution of this memo is unlimited.

Abstract

HTML+ is a simple SGML based format for wide-area hypertext documents, for use within the World Wide Web. Unlike desktop publishing formats, HTML+ captures the logical intent of authors. This simplifies the task of writing documents, and permits them to be effectively rendered on a wide range of display types as well as the printed page.

HTML+ represents a substantial improvement over the existing format: HTML, offering nested lists, figures, embedded data in foreign formats for equations etc, tables with support for titles and column headings, change bars, entry forms for querying and updating information sources and for use as questionaires for mailing. This document specifies the HTML+ format with guidelines on how it should be rendered by browsers.

Application Exhibit 23

Introduction

The World Wide Web is a wide area client-server architecture for retrieving hypermedia documents across the Internet. It also supports a means for searching remote information sources, for example bibliographies, phone directories and instruction manuals. There are three main ingredients:

a) Universal naming scheme for documents. The universal resource location syntax specifies documents in terms of the protocol to be used to retrieve them, their Internet host and path name. A format for location independent lifetime identifiers is currently being defined by working groups of the IETF. A network protocol will allow universal resource numbers (URNs) to be resolved to the URL for the nearest available copy.

b) Use of available protocols for retrieving documents over the network, including FTP, NNTP, WAIS, Gopher, and HTTP. The latter is designed specifically for use with the World Wide Web, and combines efficiency with an ability to flexibly exchange information between clients and servers.

c) A document format supporting hypertext links based on URLs and URNs which can specify documents anywhere in the Internet. HTML+ is designed for rendering on a wide variety of different display types and platforms.

Information browsers can display information in a wide variety of formats, e.g. plain text, rich text in the HTML+ format, images in the GIF and JPEG formats, MPEG movies, and MIME documents. The hypertext format has a special significance as it allows users to navigate from one document to the next at the click of a button. It provides the basis for menus, cross references, either within a document or to other documents,

---

[1]Please mail comments to the author `dsr@hplb.hpl.hp.com`, or to the WWW discussion group: `www-talk@nxoc01.cern.ch`

perhaps on the other side of the world. It also provides a means of building larger scale collections of documents that act as journals, books or encyclopedias. The format is also intended to act as a building block for creating wide area groupware applications.

HTML+ follows on from an earlier standard - HTML, see [Berners-Lee 93a], which has been widely used as the basis for hypertext documents in the World Wide Web. The new format grew out of experience with HTML, culminating in the desire to add new features, e.g. inline images, tables, and form fields for greater flexibility in querying remote information sources. This document specifies the HTML+ format and suggests ways in which browsers can choose to render it on a variety of different display types.

2. HTML+ and SGML

HTML+ itself is based on the Standard General Markup Language (SGML), an international standard for document markup that is becoming increasingly important. The term markup derives from the way proof-readers have traditionally pencilled in marks that indicate how the document should be revised.

SGML grew out of a decade of work addressing the need for capturing the logical elements of documents as opposed to the processing functions to be performed on those elements. SGML is essentially an extensible document description language, based on a notation for embedding tags into the body of a document's text. It is defined by the international standard ISO 8879. The markup structure permitted for each class of documents is defined by an SGML Document Type Definition, usually abbreviated to DTD.

Working groups in ISO have recently produced a range of SGML DTDs for documents, e.g. ISO 12083 defines DTDs for books and ISO 10744, which defines the HyTime standard for hypermedia/time-based documents. These standards are large and complex, and perhaps best suited as interchange standards that facilitate conversion between proprietary document formats. By contrast, HTML+ provides a lightweight delivery format that can be rendered by relatively simple browsers, and which has grown out of two years practical experience with wide-area hypertext information systems in the Internet community.

HTML+ and HyTime

The HyTime standard provides a rich range of architectural forms, but is not aimed at run-time efficiency. Suggestions have been made as to how the HTML DTD could be adapted to comply with HyTime's clink architectural form [Kimber 93]. This would necessitate documents declaring links as external entities and the use of local names in link definitions, but in the absence of any immediate benefit, there has been little enthusiasm for this within the World Wide Web community. Instead, it is believed that a straightforward filter program should be used to map HTML and HTML+ documents into a format which is strictly compliant with HyTime, when this becomes appropriate.

A simple example of HTML+

The following is a simple example of an HTML+ document,which illustrates the basic ideas involved in SGML.

```
<title>A Simple HTML+ Document</title>

<h1 id="a1">This is a level one header</h1>

<p> This is some normal text which will wrap at the window margin. You
can emphasise <em>parts of the text</em> if you wish. </p>

<p> This is a new paragraph. Notice that unlike title and header tags,
the matching end tag is optional.
```

The text of the document includes tags which are enclosed in <angle brackets>. Many tags have matching end tags for which the tag name is preceded by the "/" character. The tags are used to markup the document's logical elements, for example, the title, headers and paragraphs. Tags may also be accompanied by parameters, e.g. the "id" attribute in the header tag, which is used to define potential destinations for hypertext jumps.

Unlike most document formats, SGML leaves out the processing instructions that determine the precise appearence of the document, for example the font name and point size, the margins, tab settings and how much white space to leave before and after different elements. The rendering software makes these choices for itself (perhaps guided by user preferences), and so can avoid problems with different page sizes or missing fonts.

Logical markup also preserves essential distinctions that are often lost by lower level procedural formats, making it easier to carry out operations like indexing, and conversion into other document formats.

Practical experience has shown that people often make mistakes when they have to type in the markup for themselves. As a result, most browsers are tolerant of bad markup. This problem is being minimised by keeping the format as simple as possible and encouraging the development of WYSIWYG editors.

The HTML+ Document Format

The following sections go through the various features of the format with suggestions as to how browsers should render them. The DTD for HTML+ is given in Appendix I.

Parsing HTML+ Documents

By default, HTML+ documents are made up of 8-bit characters in the ISO 8859 Latin-1 character set. In future, 16 bit character sets may be used to cover a wider range of languages. The HTTP network protocol uses the MIME standard (RFC 1341) to specify the document type and the character set. It is assumed that the chosen character set includes the printable 7 bit US ASCII characters as a subset.

The DTD specifies the syntax of the document structure, in particular, which tags are permitted in any given context. Certain tags are only permitted at the start of the document. Tags and attribute names are case insensitive, thus <TITLE> is equivalent to <title>. Minimisation is forbidden to avoid problems with parsing unknown tags.

In general, SGML entity definitions are used to represent characters which would otherwise be confused with markup elements:

&	is represented by	`&`
<	is represented by	`<`
>	is represented by	`>`

Such entity definitions should be used in all places except within attribute values for tags (tag names and attribute names cannot contain these particular characters). Entity definitions can also be used for special characters, e.g. "é" for a small e with an accute accent. The full list is given in Appendix II. Additional entities may be defined within documents using the SGML entity declaration tag !ENTITY, e.g.

```
<!ENTITY sgml "Standardised General Markup Language">
```

The browser will then insert the full form whenever it comes across "&sgml;".

Repeated white space characters such as space, tab, carriage return, line feed and form feed are ignored except within preformatted text, i.e. it doesn't matter which white space characters you use or how many of them you put between words, or before or after markup elements, the effect is the same as a single space character.

It is strongly recommended that HTML+ documents start with the following external identifier, indicating that the document conforms to the HTML+ DTD. This will ensure that other SGML parsers can process HTML+ documents, without needing to include the DTD with each document.

```
<!DOCTYPE htmlplus PUBLIC "-//Internet/RFC xxxx//EN">
```

HTML+ departs slightly from pure presentation independence by allowing authors to specify rendering hints, e.g. to use a bold font for a given type of emphasis. This step was taken to give authors greater control over the final appearence, and is based upon practical experience with the earlier HTML format. In addition, attribute values are used to distinguish different subcategories of markup, rather than adding extra tags. New logical categories of emphasis etc. can be added in future without needing to change existing browsers. These decisions have made it practical to restrict HTML+ to a very small set of tags.

Backwards Compatibility with HTML

The format is designed to be largely compatible with the earlier format HTML, and it is recommended that HTML+ browsers continue support for the few tags which have been dropped. This will avoid problems for the large numbers of HTML documents without the DOCTYPE declaration. Suggestions on how to map HTML elements to HTML+ are given in Appendix III.

Normal Text

This is generally shown with a serif font and wraps on the right window margin. It can include:

- Entity references, e.g. ">" and "é"
- Significant Line breaks (the BR tag)
- Hypertext links - the A tag
- Inlined graphics or icons - the IMG tag
- Various styles of logical emphasis - the EM tag
- Embedded data in an external format, e.g. TeX equations - the EMBED tag
- Input fields for forms - the INPUT tag

Line breaks and

Line breaks have a semantic significance in some contexts, e.g. the lines of a poem or a postal address. This tag causes the renderer to start a new line at the current left margin setting. There is no corresponding end tag. The BR tag is *empty*, that is to say, it doesn't act as a container around other text or markup.

Hypertext Links

When the user clicks on a hypertext link in the document, the current document is replaced by the one referenced by the link. Links can be made to a wide range of document types, based on the URL[2] and URN[3] notations. Some document types permit links to be made to specific sections within a document[4]. The syntax for links within the same document or to documents in the same directory is particularly simple:

```
Links are defined with the <a href="#z1">A tag</a>. HTML+ supports a
number of <a href="links.html">different link types</a>.
```

In a browser this might look like:

```
Links are defined with the A tag. HTML+ supports a number of different
link types.
```

The first link is to an anchor named "z1" in the current document. The second is to a file named "links.html" in the same directory as the current document. The caption for the link is the text between the start and end tags. The value for the HREF attribute defines the destination point, and can be abbreviated in certain cases. If practical, word the caption in such a way that continues to make sense when the document is printed out. The link should be shown in a clearly recognisable way, e.g. as a raised button, or with underlined text in a particular color. For displays without pointing devices, it is suggested that a reference number is given in square brackets, which can then be typed by the user.

A more general discussion of hypertext links and their treatment in HTML+ is presented in a later section.

Inlined Graphics or Icons

These are treated like characters and inserted as part of the text, e.g.

```
This line has a egyptian hieroglyph at the end of the
line. <img src="ankh.tiff">
```

The URL notation is used to name the source of the graphics data. The *align* attribute can be used to control the vertical position of the image relative to the current text line in which the IMG element is placed. Use a value of "top", "middle" or "bottom" to align the top, middle or bottom of the image with the current text line.

---

[2]The notation for universal resource locators is defined in [Berners-Lee 93b].

[3]The notation for universal resource numbers and the protocol for resolving them to the nearest available copy is currently under study by the IETF URN working group.

[4]At the time this document was written, such links were restricted to named anchors within HTML and HTML+ documents

The *seethru* attribute allows authors to include a chromakey, i.e. a colour that designates portions of the image to be left unpainted so that the background shows through. The format for this attribute's value is dependent on the type of graphics data, and has yet to be defined.

Note that you can create simple iconic buttons, e.g.

```
<a href="bigpic.gif"><img src="littlepic.tiff"></a>
```

If the user clicks anywhere on the image, this will cause the browser to retrieve its bigger version. This approach allows users to preview images which may take significant time to download. Note that there is little additional penalty for displaying the same image at multiple points in the document. The *ismap* attribute is provided for backwards compatibility with HTML. When present the browser will send all mouse clicks and drags on the image, to the server.This mechanism is explained in more detail for the FIG tag.

Sophistocated HTML+ editors should allow authors to modify images using an external editor. Larger images should be specified with the FIG tag.which provides support for flowing text around figures, along with captions, overlays and active areas.

Various Styles of Emphasis[5]

This allows you to emphasise a portion of the text. The simplest approach is:

```
<em>default emphasis, usually shown in an italic font</em>
```

The logical role of emphasis denotes the semantic significance, e.g. a citation, or text to be input by a user for a computer program.The physical style of emphasis controls its appearence. Note that EM elements can include inlined graphics.

Logical Role of Emphasis

It is strongly recommended that the logical role of the emphasis is given with the *role* attribute, e.g.

```
<em role="cite">a citation</em>
```

Providing a logical role allows browsers to apply differing rendering styles according to the role, but more importantly, it allows indexes to be constructed automatically, e.g. the list of bibliographic references in a technical report. These can be used for searching through collections of documents according to semantic keys giving better focussed searches compared with full text indexes .

The list of recommended roles are as follows: *(this can be given in upper or lower case)*

For references to other works:

cite	a reference to a related work
pub	a publication containing a referenced work
author	an author of a referenced work
editor	an editor of a referenced work
title	the title of a referenced work
credits	e.g. the rights owner of a photograph
copyright	the holder of the copyright
isbn	for ISBN numbers
acronym	for acronyms like "NATO" and "US"
abbrev	for abbreviations

For annotations:

footnote	shown as footnote or pop-up
margin	shown as margin note or pop-up

For computer instruction manuals:

dfn	defining instance of a term

[5]The name EM was chosen in preference to EMPH because it allows existing HTML browsers to show all HTML+ emphasis in italics. It also allows HTML+ browsers to correctly process the common case for emphasis in HTML documents.

kbd	something a user would have to type
cmd	command name, e.g. "chmod"
arg	command arguments, e.g. "-I"
var	named place holder, e.g. "filename"
ins	an instanced of a named printer, directory or file etc.
opt	an option of some kind
code	an example of code (shown with a fixed pitch font)
samp	a sequence of literal characters

On dumb terminals annotations should be shown in round brackets. Margin notes should be right aligned, and may include graphics via the IMG tag. The set of recommended roles will be kept by the HTML+ registration authority.

Physical Styles

The appearence can be modified by adding optional rendering hints from the list:

```
<em b>              bold text
<em i>              italic text
<em u>              underlined text
<em sup>            superscript text
<em sub>            subscript text
<em tt>             type writer font (courier)
<em hv>             sans serif font (helvetica)
<em tr>             serif font (times roman)
```

These hints can be combined, e.g.

```
<em b i> for bold italic text </em>
```

Note that these are only hints and may be ignored by browsers. Indeed, arbitrary combinations will present difficulties for most browsers. If the display is limited to a single font, colour or underlining can be used, but should be clearly differentiated from hypertext links and headers. Dumb terminals can use email conventions, e.g. switching to all capitals, or delimiting with the * or _ characters. Subscript and superscript text should be shown in a smaller point size, vertically offset as appropriate.

Browsers may choose to simplify or ignore hints, but should aim to do so in a consistent manner. At the simplest level, browsers can ignore the attributes and render all emphasis in the same style.

Nested Emphasis

Emphasis can be nested as in:

```
<em b>bold text, and <em i>bold italic text</em></em>
```

Nested emphasis is better suited for grouping logical roles together, for instance, you could use EM to separately tag author, title, and publication, and then wrap these up as a citation. Without this, indexing programs will have difficulty in grouping markup into the correct references.

Horizontal Rule

The <HR> tag may be used to draw a horizontal rule to separate text sections. It can be rendered as a simple line across the middle section of the window/paper or similar decoration.

Embedded data in an external format

The EMBED tag provides a simple form of object level embedding. This is very convenient for mathematical equations and simple drawings. It allows authors to continue to use familiar standards, such as *TeX* and *eqn*. Images and complex drawings are better specified using the FIG or IMG elements. The *type* attribute specifies a registered MIME content type and is used by the browser to identify the appropriate shared library or external filter to use to render the embedded data, e.g. by returning a pixmap. It should be possible to add support for new formats without having to change the browser's code, e.g. through using a common calling mechanism and name binding scheme. Sophistocated browsers can link to external editors for creating or revising embedded data. Arbitrary 8-bit data is allowed, but &, < and > must be replaced by their SGML entity definitions. For example <embed type="application/eqn"> 2 pi int sin (omega t)dt </embed> gives

$$2\pi\int \sin(\omega t)dt$$

Input Fields for Forms

Input fields can be arranged with considerable freedom, as part of normal paragraphs, preformatted text, lists or tables. Examples of how to do this are given later on in the section describing the FORM tag. The INPUT tag has the following attributes:

name Used to name this input field, e.g. `name="phone number"` *(required attribute).*

type Defines the type of data the field accepts (the type name is insensitive to upper/lower case). If missing, the field is assumed to be a free text field.

size Specifies the size/precision of the input field according to its type, see below *(optional).*

value The initial value for the field, or the value when checked for checkboxes and radio buttons *(optional, except for radio buttons).*

checked When present, this attribute indicates that a checkbox or radio button is selected.

disabled When present, this attribute indicates that this field is temporarily disabled. Browsers should show this by greying out or via a similar visual clue. Users are unable to set the focus to disabled fields, or change their values.

error When present, this attribute indicates that the current value for this field is in error in some way, e.g. because it violates some consistency constraints. Browsers should indicate this by a change to the shape and colour (red) of the field's border. This should be accompanied by an error message and a beep.

The following types of field should be supported: *(in either upper or lower case)*

text Single or multi-line text entry fields. Use the *size* attribute to specify the width and height in characters, e.g. size="24" or size="32x4".

url For fields which expect document references as URL or URNs.

int For entering integer numbers, the maximum number of digits may be given with the *size* attribute, e.g. size=3 for a 3 digit number[6].

float For fields restricted to floating point numbers.

date Restricted to a recognised date format.

checkbox Use these for simple boolean attributes, or for attributes which can take multiple values at the same time from some set of alternatives (for fields with the same *name*).

radio Use these for attributes which can take a single value from a set of alternatives (groups input fields with the same *name*).

For the purposes of sending the contents of a form to a server, as part of a query, the input fields are mapped to a list of properties. In most cases the *name* and current *value* are used to define a property/value pair for each field. Radio buttons and check boxes are left out of the list if they are unselected. This ensures that only the selected radio button yields a property/value pair. By missing out the *value* attribute for check boxes, these fields will map to a simple (value-less) property. The representation of property lists is defined as part of the HTTP protocol.

Browsers can choose to notify the server whenever a field is changed (i.e. when a field looses the focus and its contents have changed) or wait until the form is completed. This choice will depend on network latency.

Drop down or "combo" style selection lists may be added in a future revision to this standard.

---

[6]Perhaps the syntax should permit integer ranges, e.g. size="1 to 6", in which case a more appropriate name for the attribute than *size* would be desirable.

OCT 24 2003
PATENT & TRADEMARK OFFICE

Headers and Titles

The title tag is generally used to define the window banner when viewing a particular document, e.g.

```
<title>Reference Guide to HTML+</title>
```

This element should appear at the start of the document. There are six levels of headers, H1 to H6, with H1 the most important, and H6 the least. A common convention is to begin the body of the document with a level one header. e.g.

```
<h1>Introduction to HTML+</h1>
```

Header names should be appropriate to the following section of the document, while the document title should cover the document as a whole. There are no restrictions on the sequence of headers, e.g. you could use a level three header following a level one header. Browsers should render headers with a line break before and after the header text. A common convention for headers is to use a sans serif font, e.g. Helvetica, with a smaller point sizes for less significant headers, and a serif font, e.g. Times Roman, for normal text.

Headers can include an identifier, unique to the current document, for use as destinations of hypertext links, e.g.

```
<h1 id="intro">Introduction to HTML+</h1>
```

This allows authors to make links to particular sections of documents. It is a good idea to use something obvious when creating an identifier, to help jog your memory at a later date. WYSIWYG editors may automatically generate the identifiers. In this case, they should also provide a point and click mechanism for defining links, so that authors don't need to deal explicitly with the identifiers.

The attribute "margin" when present acts as a hint to the browser to insert the header into the margin and causes the following text to be vertically aligned with the start of the margin header. By convention, margin headers are left justified, e.g.

```
<h4 margin> Deleting the Curve </h4>

The Delete command allows you to delete any selected symbol or text
block.
```

Note that headers don't act as containers for the subsequent text. You can group the header and text with the GROUP tag, see later for details.

Indexing

A good index plays an important role in helping you find your way to the material you need. It allows you to type in one or more keywords to see a list of matching topics. Alternatively you can browse through the index and take advantage of serendipity. This also allows you to gain a feeling for the limits of what is covered. The two approaches can be combined, when the characters typed act dynamically to control the viewing position within the index.

Typically each keyword entry in the index is associated with one or more topics. This notion of guiding the user is absent from full text indexes like WAIS, where users are given very little help in choosing the keywords to search on. Generating a conventional index for a document is a skilled task, and HTML+ allows authors to include directives for creating an index. These directives can be included with document titles, headers and emphasis etc. using the *index* attribute. This allows each such element to be included in one or more entries in the index, under primary or secondary keys, e.g.

```
<h3 id="z23" index="Radiation damage/shielding from as difficult">Radiation shielding</h3>
```

This resulting index looks like:[7]

Radiation damage
  classical target theory
  dominance of
  in molecular mills
  shielding from as difficult
  simple lifetime model
  track-structure lifetime model
Radicals
  and so on.

Where each entry is a hypertext link to the associated anchor. The *index* attribute can specify multiple entries, each separated with the ";" character. The optional secondary key (*shielding from as difficult*) is introduced by the "/" character. Secondary keys are useful when the primary key occurs more than once. To allow for future extension, primary keys should not start with the "#" character. This prefix is being reserved to designate indirect index entries. Use "\/", "\;", "\#" and "\\" to escape "/", ";", "#" and "\" respectively.

Paragraphs and Preformatted Text

HTML+ includes support for paragraphs and preformattted or verbatim text.

Defining Paragraphs with <P>

The P tag is used to define paragraphs. Unlike many other tags, the end tag is optional. Note, that unlike HTML,the tag acts as a container for the text of the paragraph. This allows paragraphs to act as hypertext anchors. The end of the paragraph is implied by finding markup elements which are not permitted as part of a paragraph.

The following attributes may be used:

id	An identifier, unique to this document, which can be used as a destination in a hypertext link.
role	The role of the paragraph, see the following list for supported types.
align	A rendering hint to the browser to justify lines. The supported values should be: `align="left"`, `align="center"`, `align="right"` and `align="flush"`. This is useful for single line paragraphs or when the lines are made explicit with the tag.
indent	When present, this hint suggests that the left and right margins are indented by an amount dependent on the browser, e.g. about 4 character widths.

The *role* attribute is used to indicate the logical role of the paragraph, e.g. a stanza in a poem or a cautionary note in a computer manual. Browsers may apply particular rendering styles to certain roles. The role name is case insensitive.

The following roles are recommended: *(in upper or lower case)*

quote	A paragraph quoted directly from some other work. Browsers could indent the paragraph and maybe use a different font.
byline	Information about the author of the document, e.g contact details. This could be displayed in a different font, and perhaps right aligned.
note	Advisory note in an instruction manual. The browser could display a hand icon in the margin.
caution	Cautionary note. The browser could display an warning road sign in the margin.
error	A note describing error conditions. The browser could indicate the importance of the note by displaying a stop sign in the margin.

[7]Taken from K. Eric Drexlers's "Nanosystems, Molecular Machinery, Manufacturing and Computation".

An example of a paragraph element:

```
<p role="note"> If you accidentally delete a symbol other than the red
circle, immediately press ALT+BKSP to choose the undo command, and
then select the red circle and delete it again.
```

Paragraphs can be rendered by indenting the first line, or by leaving a vertical gap, for example, half the current line spacing. When using the latter style, browsers should take care to avoid inserting this vertical gap when the paragraph element immediately follows a header. This rule ensures that authors can tag paragraphs directly following a header without causing unwanted extra space to appear before the start of the text.

Preformatted Text with <PRE>

This is generally shown in a fixed pitch font and preserves the original spaces and line breaks. The horizontal tab character is deprecated, but should be interpreted as a move to the next tab setting, at every eighth column. Preformatted text is useful for including plain ASCII text, e.g. program listings and email messages. A number of tags can be included within preformatted text, e.g. hypertext links using the A tag, emphasis, inline images and input fields. The following optional attributes can be used:

id	An identifier, unique to this document, which can be used as a destination in a hypertext link. Note that the paragraph tag acts as a container for the paragraph.
role	The role of the element.
tr	Use a proportional serif font, e.g. Times Roman.
hv	Use a proportional sans serif font, e.g. Helvetica.
width	This gives the maximum number of characters which will occur on a line. The default value is assumed to be 80. Browsers recognising this attribute should optimally handle widths of 40, 80 and 132, with other widths being rounded up.

Preformatted text started off in HTML with a simple mechanism for showing computer output, for which the spaces and line breaks were significant in determining the layout. The desire to render Unix manual pages as hypertext forced a rethink. The new version supported character emphasis and hypertext buttons for cross references. HTML+ adds the capability to use variable pitch fonts, and to set up tab stops, e.g.

```
<tab at=40 align=right>
```

The *at* attribute specifies the position of the tab stop, measured from the left margin in terms of the width of the character M for the current font. The *align* attribute is one of "left", "center", "right" or "decimal", defaulting to left alignment. For greater control of layout, authors should exploit the FIG or EMBED tags to use external formats, for example encapsulated Postscript. Unfortunately these formats don't as yet support hypertext links.

Ordered, Unordered and Definition Lists

There are three kinds of lists: ordered or numbered lists, unordered lists and definition lists. Ordered and unordered lists can be nested arbitrarily, and browsers should progressively inset the left margin for each level of nesting.

Ordered Lists with <OL>

The list items are automatically numbered, e.g.

```
<OL>
  <LI>Wake up
  <LI>Get dressed
  <LI>Have breakfast
  <LI>Drive to work
</OL>
```

Is displayed as:

1) Wake up
2) Get dressed
3) Have breakfast
4) Drive to work

The *compact* attribute when present, e.g. <ol compact>, has the effect of reducing interitem spacing. Authors can also make both the OL tag and the LI tag potential destinations for hypertext links with the *id* attribute. List item text can include normal text and paragraph elements, but not headers.

Unordered Lists with <UL>

These are bulleted lists, e.g.

```
<UL>
  <LI>Wake up
  <LI>Get dressed
  <LI>Have breakfast
  <LI>Drive to work
</UL>
```

Is displayed as a bulleted list:

- Wakeup
- Get Dressed
- Have breakfast
- Drive to work

The *compact* attribute when present, e.g. <ul compact>, has the effect of suppressing bullets and reducing interitem spacing. Multicolumn lists can be requested with the *narrow* attribute, e.g. <ul narrow>. This causes the browser to try to lay out the list as a number of columns, depending on the window width. This attribute should only be used when all the items are less than 20 characters long. Authors can also make both the UL tag and the LI tag potential destinations for hypertext links with the *id* attribute. List item text can include normal text and paragraph elements, but not headers. For nested unordered lists, browsers may use different bullet symbols for different levels, in addition to progressively insetting the left margin. The *src* attribute on the LI tag can be used to specify an icon for use in place of the standard bullet symbols.

Definition Lists with <DL>

These consists of pairs of terms <DT> and definitions <DD>. The following example is part of a french dictionary:

```
<DL>
  <DT>endetter
  <DD>Engager dans des dettes

  <DT>endeuiller
  <DD>Plonger dans le deuil, remplir de tristesse

  <DT>endiablé, ée
  <DD>D'une vivacité extrême
</DL>
```

Is commonly displayed as:

endetter	Engager dans des dettes
endeuiller	Plonger dans le deuil, remplir de tristesse
endiablé, ée	D'une vivacité extrême

With the *compact* attribute, e.g. <dl compact>, this could be altered to:

endetter Engager dans des dettes

endeuiller Plonger dans le deuil, remplir de tristesse

endiablé, ée D'une vivacité extrême

In this style, the term and definition appear in the same paragraph, with the term text emphasised in a bold font. The definition text follows on, and wraps to a left margin a little further inset than the term text. This style is common place in dictionaries.

Term text following the <DT> is restricted to normal text. The definition text after the <DD> tag can additionally include paragraph elements and ordered/unordered lists. Headers are not allowed in either case. Authors can make the DL, DT and DD tags potential destinations for hypertext links with the *id* attribute.

Authors are reminded to check that DT and DD are paired up. Common misunderstandings lead to people repeating DD tags to separate paragraphs (use <P> instead), or leaving out the DT tag altogether to indent text (use <p indent> or <group indent>). The ability of browsers to cope with bad markup seems to encourage such problems, which will hopefull fade away as wysiwyg editors become commonplace

Figures

Figures provide great flexibility, and can be used to show images, graphics or other information specified in an external format:

- linked or embedded definitions
- control of picture alignment and text flow
- Figure description for when the image can't be shown
- caption placement
- scaled or pixel-based coordinates
- hypertext links with active areas
- text and image overlays

The following simple example will set the scene for the description of the various features:

```
<fig align="right" src="map.gif"> How to get to my house </fig>
```

Here, the image is defined by a link to an external document. The caption "How to get to my house" will appear at the bottom of the image. The *align* attribute directs the browser to display the figure at the right of the widow, and to flow subsequent text around the left of the image.

Using embedded graphics data

Instead of the *src* attribute, you can include an EMBED element immediately following the <fig> tag. This is useful for simple graphs etc. defined in an external format.

Figure Description

The FIGD tag allows you to give a textual description which can be shown when the figure itself can't be shown, e.g. for browsers working on dumb terminals, e.g.

```
<FIGD> This is an aerial photograph of central London, showing
Buckingham Palace and the Houses of Parliament. On the left you can see
Hyde Park and in front the Albert Hall and the Natural History
Museum.</FIGD>
```

Alignment and Text Flow

The *align* attribute controls the horizontal position of the figure: "left", "right", or "center". The default is "left". Browsers may flow text when there is sufficient room, unless the figure is center aligned or the *noflow* attribute is present.

Caption Placement

The *cap* attribute allows you to ask the browser to position the caption text to the "left", "right", "top" or "bottom". The default is to place the caption at the bottom of the figure. Text flow will occur around the figure and caption, leaving a suitable gully. The browser will ignore this attribute if there is insufficient room for the requested placement.

Pixel-base or Scaled Coordinates

The upper left of the figure is designated as x,y = (0, 0), with x increasing across the page, and y down the page. If points are given in real numbers, the lower right is taken as being (1.0, 1.0), otherwise with integer values, the coordinates are assumed to be in pixels[8]. Note that using scaled coordinates is much safer, especially for graphics! The extent of the image in pixels may change, e.g. as a result of format negotiation with the server, and by retrieving images with lower resolution when network performance is poor.

Active areas

The *ismap* attribute causes the browser to send mouse clicks on the figure, back to the server using the selected coordinate scheme. The mouse button-up event is sent with the URL formed by adding "?x,y" as a suffix to the URL for the current document. You can also designate rectangular regions of interest in the picture by holding the mouse button down while dragging the mouse. The browser should show a rubber band outline for the rectangle defined by the current location of the mouse pointer and the point at which the mouse button was pressed. The region is named by taking the current URL and adding the suffix: "?x1,y1;x2,y1;x2,y2"[9], where (x1, y1) and (x2, y2) define the points at which the mouse button went down and came up, respectively. The *ismap* mechanism is relatively slow, but makes sense when the active regions change their boundaries over time, e.g.

```
<fig ismap src="weather.gif">Click on your area for todays weather</fig>
```

You can also designate arbitrary areas of the figure as hypertext links. Mouse clicks are handled locally, and the browser can provide visual clues that the pointer is over an active area, for example, by changing the pointer from an arrow to a hand symbol, or highlighting the area in some way.

Active areas are defined with the FIGA tag. This has two attributes:

href	A URL specifying the link to traverse when clicked (required)
area	Defines a polygonal[10] area as a list of points: "x1, y1; x2, y2; ..." (optional)

The *area* attribute lists a sequence of points defining a polygon. Closure is ensured by joining the last point in the list to the first (i.e. a triangular area is defined with a list of 3 points).When the *area* attribute is missing, the whole of the picture is assumed. Polygons may be non-convex or even intersect themselves, thereby complicating the definition of what is enclosed by the polygon. Holes should be excluded. Note that active areas defined with FIGA take precedence over the *ismap* mechanism.

Overlays

The FIGT tag allows you to position text and image overlays on top of the figure, e.g.

```
<fig src="map.giff">
  <figt at="0.2, 0.3" framed>A text overlay</figt>
  The figure caption
</fig>
```

---

[8]This mechanism was designed to be backwards compatible with the *ismap* feature as used with IMG in HTML, and as a consequence forces the choice of y increasing down rather than up the page. A simple test to distinguish the two schemes is to check if the "." character occurs anywhere in the list of points.

[9]This definition is intended to allow future extension to arbitrary polygons, and hence is chosen to be directly compatible with the *area* attribute of the FIGA tag.

[10]The code for hit testing polygons is tricky, but quite fast. A public domain version of the code would be helpful.

The overlay can contain a wide variety of elements including text, images (IMG), lists and tables. Figures shouldn't be nested. Any hypertext links in the overlay text will take precedence over the *href* attribute in FIGT. The following attributes are permitted:

idref Names the FIG element to which this overlay applies. This is only relevant when overlays are held separately as a form of annotation. This attribute then allows the annotation to be correctly merged back into the document.

at The upper left of the overlay, relative to the figure.

width As a fraction of the figure, e.g. width="0.3". This allows you to limit the lengths of wrapped text lines. The vertical extent is then determined automatically.

framed Directs the browser to draw a frame around the overlay and to colour in the background in some way.

href Allows you to make the overlay into a hypertext button.

Tables

Tables are defined with the TBL tag. Cells are designated as being headers or data. You can join adjacent cells, e.g. to define a header spanning two columns.

An Example of a Table

	average		other
	height	weight	category
males	1.9	0.003	yyy
females	1.7	0.002	xxx

This is defined by the markup:

```
<tbl border>
   <tt top> An Example of a Table
   <th rowspan=2> <th colspan="2"> average <th> other <tr>
   <th> height <th> weight <th> category <tr>
   <th align=left> males <td> 1.9 <td> .003 <td> yyy <tr>
   <th align=left> females <td> 1.7 <td> .002 <td> xxx
</tbl>
```

The *border* attribute for TBL directs the browser to draw borders. The *compact* attribute is used when you want the table to appear in a smaller size.

The optional <tt> tag defines a title. By default (i.e. when *top* is missing) this should be positioned below the table. The <th> and <td> tags define header or data cells respectively. The <tr> tag acts as a separator between rows. In the example, you can see that the first header in each of the first two rows is void.

TH, and TD all have the same permitted attributes:

colspan Columns spanned by this cell, see example

rowspan[11] Rows spanned by this cell, see example

align=left Left justify the cell's content

align=center Center justify the cell's content

align=right Right justify the cell's content

[11]This is tricky to handle. The parser should carry a spanned cell over to the next row, the definition of which should miss out the spanned cell, i.e. the next row will have one fewer explicit cell definitions.

By default, headers are centered, while other cells are left justified. If practical, browsers should be smarter than this, e.g. if all the cells in a column are shorter than the column header, then indent the cells to make them appear under the middle of the header.

Browsers need to carry out a pre-parse (e.g. when sizing the vertical scroll bar) in order to determine the number of columns and their widths. The following guidelines may be useful:

- There is no need to declare empty cells at the end of a row, so the number of columns for the table is given by the row with the most columns.
- Restricting text to a fixed pitch font may simplify matters.
- If a column only contains numbers or empty cells then align on units and set width to the maxium precision needed (before and after decimal point, allowing for an exponent). This rule also applies when currency symbols are used.
- Otherwise set column width to the minimum of a threshold width and the maximum text length for all cells in the column. Text is left aligned and wrapped if it exceeds the chosen column width.

The threshold column width can be set according to the number of columns and the width of the display window. It is also necessary to take the column headers into account in this process. Header text wraps to the next line if the column is too narrow. Browsers will by default center the header in the column.

A complication occurs when a header or data cell spans more than one column, as specified by the *s* attribute. This can be used to give complex headers which share a header between columns followed by individual headers on the next line.

Vertical gaps can be introduced with the <tb> element - this inserts 1/2 line space into the next row. Header and Data rows can be intermixed. Authors can use alternate header and data rows when the rows alternate between text and numbers. The vertical alignment of numbers only applies to data fields.

Tables which don't fit into this model should be defined as figures using an external format, e.g. Postscript, TeX or Computer Graphics Metafile.

Forms

A document can include one or more forms. Each form is defined by a FORM element, which contains a number of input fields laid out with normal and preformatted text, lists and tables. The browser should manage the input focus, e.g. with the tab key and mouse clicks. The Return key can be used to mean that the user has filled in the form and wants the appropriate action to be taken. Browsers may also display "Accept" and "Cancel" buttons as part of the document (or perhaps on another part of the browser). Note that forms shouldn't be nested.

The action to be taken is specified by the *action* attribute of the FORM tag. If missing the URL for the current document is assumed. This attribute uses a URL to specify a server to query, or an email address to send the form to. When sending the form to a server as a query, the form's contents are encoded as a property list (see definition of the INPUT tag). The precise encoding is dependent on the HTTP protocol and defined in [Berners-Lee 93c][12]. When the form is to be mailed, it is first converted into plain text, closely resembling the appearence on the screen. You can include multiple RFC 822 mail headers with the MH tag. The *hidden* attribute may be used to hide the headers when browsing the document. The following is an example of a simple questionaire:

```
<form action="mailto:www_admin@info.cern.ch">
<mh hidden>
  Subject: WWW questionaire
</mh>
Please help us to improve the World Wide Web by filling in the
following questionaire:
<p>
Your organisation? <input name="org" size="48">
```

[12]This and the *ismap* feature rely on the forthcoming definition of HTTP as an official Internet standard.

```
<p> commercial? <input name="commerce" type="checkbox">
How many users? <input name="users" type="int">
<p> Which browsers do you use?
<ol compact>
<li> X Mosaic <input name="browsers" type="checkbox" value="xmosaic">
<li> Cello <input name="browsers" type="checkbox" value="cello">
<li> Viola <input name="browsers" type="checkbox" value="viola">
<li> Others? <input name="other browsers" size="48x4">
</ol>
A contact point for your site: <input name="contact" size="48">
<p>Many thanks on behalf of the WWW central support team.
</form>
```

Floating Panels

The PANEL tag can be used to define panels or boxes which are free to float with respect to the standard flow of text. These are often used in magazine articles for asides on background material. The panel is typically shown with a distinctive background colour and border. The lay out software positions the panel to coincide with the page boundaries in printed media. For on-line use, panels can be rendered as pop-up windows. The body of the panel can be defined by a link to a separate document or included in the current document.

Another application of the panel tag is for annotations by one or more reviewers. The annotations can be held separately e.g. with an annotation server and subsequently retrieved and merged with the document. The annotation server returns a list of annotations in response to being queried with a URL or URN. This list can be expressed as a multi-part MIME message with the author and date passed as RFC 822 headers for each annotation.

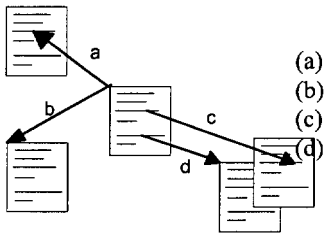
The following optional attributes are permitted with the <panel> tag:

id	An identifier, unique to this document, which can be used as a destination in a hypertext link.
at	An identifier elsewhere in this document. The panel mustn't be placed before this point. (Defaults to the current position if the *at* attribute is missing).
role	This may be used to clarify the role of the panel, e.g. as an "annotation" or an "aside".
href	This attribute allows authors to fill the panel from a separate document, as specified by a URL. Note that the matching end tag: </panel> is always needed.

The text contained by the panel element can include any of the markup elements and looks like a separate document (panels themselves can't be nested). If the *href* attribute is used the text delimited by <panel> ... </panel> may be used as the caption for a pop-up. The *at* attribute allows you to include the panel definition at a convenient point in the HTML+ document, rather than interrupting the main flow of the document.

More on Links

Before describing the details of how links are represented in HTML+ it is worth looking more generally at the nature of hypertext links. First a terminological point: a *node* is the atomic unit for information retrieval, while *documents* may consist of one or more nodes, perhaps arranged as a hierarchy. A node may even be shared between several documents. Hypertext links start and end on nodes or anchors, where anchors specify portions of nodes, e.g. a paragraph or list.



The diagram illustrates the basic possibilities:

(a) Link from a node to an anchor
(b) Link from a node to a node
(c) Link from an anchor to another anchor
(d) Link from an anchor to a node

Links in HTML+ are represented with the LINK and A tags. The LINK tag is used for cases (a) and (b), while the A tag is used for cases (c) and (d). These links are held in the source node only, so there is a risk that the destination may have disappeared. Organisations can manage this risk by long term support for a few well published nodes (servers can use redirection to hide internal name changes for these nodes). Links to other subsidiary nodes are at higher risk. This structured approach allows people to become familiar with the major routes through the web, without needing to worry about the minor routes.

In most cases URLs and URNs explicitly specify a node/anchor. The nodes may be explicit files or generated as the result of some process invoked by the server, e.g. a hypertext listing of a directory or a list of matches for a given search string. The search string can be explicitly encoded as part of a link, or dynamically defined by the user (see the ISINDEX tag, as described later on).

Links may be held separately from the source and destination nodes[13]. This is particularly appropriate for annotations and discussion groups. For example, consider making an annotation on a document held by a server located far away in another organisation. You could take a local copy and directly annotate it, but this is only appropriate for private use. The remote server might even support a protocol to add your annotations in place. More likely though, you will have to use an annotation server. This mechanism can be used to obtain a copy of the document with the annotations inserted as hypertext links and shown as pop-ups or separate documents.

Context Dependent Links

For discussion groups, responses are made asynchronously, and include one or more references to other articles. In this situation, context dependent links are appropriate. The resolution to an explicit node can be carried out by either the client or server. The former approach is often appropriate, but requires special support in the browser, e.g. for network news and nntp.

Context dependent links are also useful for links to the table of contents for documents consisting of multiple nodes, when some of the nodes also appear in other documents. The appropriate table of contents for a given node will depend on which document is currently being viewed. In this case, the context will depend on how the current node was reached.[14] This is quite simple to track if the links from the table of contents are differentiated from cross reference links.

Hypertext paths are recommended routes through a set of nodes, and generally shown by next and previous buttons on a toolbar. Paths can be defined using explicit links in a node, or held separately in another node. The latter case once again, depends on the context. Paths and tables of contents all fall under the general category of navigating around a hierarchy of nodes forming a document too large or unwieldy to be held in a single node.

Types of Links

There are several motivations for differentiating between types of links:

how it is viewed	The potential to show different cues depending on the type and size of the node to be retrieved. If this information is explicitly stated as part of the link, there is a chance that it will become out of step with the linked node.
what happens	Whether the linked document replaces the current one, or appears in a new window, or as a pop-up overlay on top of the current one.
printed appearance	Whether links are treated as references, footnotes or as separate sections
effect on context	After traversing the link, will there be implicit values for the table of contents, and hypertext path etc?

[13]These correspond to HyTime's *ilink* architectural form.

[14]This is more general than deriving the role of the link from that of the node alone.

Link Attributes

The A tag has the following attributes:

id	An identifier unique to this document which can act as a hypertext anchor
name	The same as *id* and included for backwards compatibility with HTML. New documents should use the *id* attribute for consistency with the other tags.
href	The URL or URN identifying the destination of the link.
role	A string giving the role of the link, e.g. role="partof" or "annotation"
effect	A string defining how the linked node is shown: "replace", "new", "overlay", with the default effect of replacing the current document.
print	How should the link be printed: "reference", "footnote" and "section", defaulting to "reference" (i.e. a footnote stating the link's URL).
title	The title to show when otherwise undefined for the node.
type	The MIME content type for the linked node for use with presentation cues.
size	The size in bytes for the linked node. This allows the browser to show a gauge indicating progress in retrieving long documents or images etc.

The LINK tag has only the *href* and *role* attributes.

The *role* attribute is appropriate when context dependent properties such as *table of contents* (toc) are implied for the linked node, e.g. if the current node is a toc (as defined by the *html* or *group* tags) and the link has the role "partof", then the current node should act as the *toc* for the linked node. This property propagates down "partof" links, but not normal links. The *next* and *prev* properties are given by the sequence of "partof" links in the parent node. The *parent* property is only defined if the current node was reached via a "partof" link.

The LINK tag is used to express these properties in an explicit form, e.g.

```
<LINK href="toc.html" role="toc">
```

The recommended property names are: (in upper or lower case)

toc	Table of contents for current node.
next	The next node in a hypertext path.
prev	The previous node in a hypertext path.
parent	The next level up in the hierarchy.
index	A searchable index appropriate to this node.
style	The style sheet appropriate to this node.

Style sheets provide a way for authors to express their detailed preferences for fonts, and layout, whether for the screen or when the node is printed out. A possible format is given in [Raisch 93].

The *effect* attribute is a hint and may be disregarded by browsers. It allows you to click on an image and to see a linked movie as an overlay at the same position. The browser tries to position the overlay at the same origin as the link. In some cases, the linked node is a description of the current node. By including `effect="new"`, the linked node will appear in a new window so that users can see both nodes at the same time. This hint should be used sparingly!

The *print* attribute makes it practical to print nodes along with relevant linked nodes. By default each link appears as a footnote stating the link's URL. Short nodes can be included in their entirety as footnotes, and longer ones as sections in their own right. This approach could be extended in future, to reorder the sequence of nodes from that defined by the position of the links in the source node, and to control the level that nodes appear as, e.g. chapter, section or subsection.

The *title* attribute is useful for nodes without titles of their own, e.g. Gopher menus. The *type* attribute can be used to show cues for the node type, e.g. iconic decorations[15]. The *size* attribute allows browsers to show a gauge on how much of a document has been retrieved at a any time. These attributes are liable to get out of step with the target node, and should be treated as hints only.

Groups

The GROUP tag allows you to define arbitrary groups, e.g. books, chapters, and sections. The *role* attribute is used to name the logical role of the group. You can use most markup elements inside a group element, including group itself. The *inset* attribute is a rendering hint to inset the left margin. Using the A tag with role="partof" allows you to designate a node as being included within the group, allowing hierarchies of groups which cross multiple nodes. See previous discussion of how properties are propagated.

Groups offer opportunities for presenting and searching documents at different levels of abstraction. For example, you might first describe a book by its title, author, publisher and ISBN number. The next level down could add a cover illustration together with a summary of the book's contents, some comments by reviewers and a short biography of the author. This would allow a list of books to be presented in an iconic form using a miniature version of the "cover page". Publishers could include copyright and other details in a standard place.

Change Bars

Authors can indicate a part of a document has been changed using the CHANGED tag. This may appear anywhere that normal text is allowed (as designated by the entity reference `%text;` in the DTD). This tag signals the beginning or end of changes, which should be rendered by a vertical bar in the left margin. The tag can have one (but not both) of the following attributes:

id An identifier unique to the current document, which can also be used as a a destination for hypertext links. This signals the beginning of changes, e.g. `<changed id=z34>`.

idref This must be an identifier matching the preceding changed element. It signals the end of changes. Note that you mustn't have both *id* and *idref* together, e.g. `<changed idref=<z34>`.

Notes

The NOTE tag allows authors to name an arbitary portion of a document. It can be used much more freely than the A tag which is restricted in the markup it can contain. The NOTE tag is intended for delimiting the portion of a document associated with an annotation. The annotation itself is linked via the *href* attribute or expressed as a panel element, whose *at* attribute names this note. The latter allows several annotations to apply to the same place.

The permitted attributes are:

id An identifier unique to the current document, which can also be used as a a destination for hypertext links. This signals the beginning of the note, e.g. `<note id=z34>`.

href May be used to directly specify an associated annotation with a URL.

idref This must be an identifier matching the preceding note element. It signals the end of changes. Note that you mustn't have both *id* and *idref* together, e.g. `<note idref=z34>`.

---

[15]The appropriate cue might also depend on the role of the link, e.g. for annotations browsers could show an icon of a drawing pin (as in attaching a note to a pin board). The colour of the pin could then vary according to the media type of the annotation.

Miscellaneous Tags

The remaining tags must appear at the start of the node like TITLE and LINK. They describe properties which apply to the node as a whole.

The HTML tag.

This is intended to provide short informal classifications for use in cataloging documents held by HTTP servers. The *role* attribute identifies the purpose of the node, for example `<html role="home page">`. Another common role is "toc" for table of contents. See previous discussion of link attributes.

The ISINDEX tag

This specifies that the URL designated with the *href* attribute is searchable (defaults to this document's URL). Browsers should allow users to enter a search string of one or more keywords. When the Return key is pressed the search string is appended to the designated URL, after a "?" character and sent to the server specified by the URL. Certain characters should be escaped as specified by the standard URL syntax, for example, the space character is mapped to "+". The newer HTTP protocol offers an alternative means for specifying that documents are searchable. See [Berners-Lee 93c] for details.

The NEXTID tag

This is used by browsers that automatically generate identifiers for anchor points. It specifies the next identifier to use, to avoid confusion with old (deleted) values, e.g. `<nextid n="id56">`. The identifier should take the form of zero or more letters followed by one or more digits. The numeric suffix should be incremented to generate successive identifiers.

The BASE tag

The *href* attribute gives the full URL of the document, and is added by the browser when the user makes a local copy. Keeping the original URL in a local copy is essential when subsequently viewing the copy as it allows relative URLs in the document to be resolved to their original references.

Note that one motivation for using relative URLs is to allow a group of documents to be copied without the need to alter any links between them. In this case, the BASE tag is inappropriate, since it would cause links to be intepreted as being to the original documents rather than their copies.

The HEAD and BODY tags

The HEAD tag can be used to delimit properties which apply to the document as a whole, and if used, must be present at the start of the document, followed by the BODY tag which then delimits the rest of the document.

http://www.w3.org/pub/WWW/MarkUp/HTMLPlus/htmlplus_1.html

Acknowledgements

I would like to thank the many people on the *www-talk* mailing list who have contributed to the design of HTML+ and to the management of HP Labs for their support during this work.

David Raggett, Hewlett Packard Laboratories, July 1993.

Email: dsr@hplb.hpl.hp.com, *Phone*: +44 272 228046

OIPE JC64
OCT 24 2003
PATENT & TRADEMARK OFFICE

Appendix I - The HTML+ DTD

```
<!SGML  "ISO 8879:1986"
--
      Document Type Definition for the HyperText Markup Language
      Plus for use with the World Wide Web application (HTML+
      DTD).
      NOTE: This is a definition of HTML+ with respect to
      SGML, and assumes an understanding of SGML terms.
--
CHARSET
      BASESET     "ISO 646:1983//CHARSET
                  International Reference Version (IRV)//ESC 2/5 4/0"
      DESCSET     0     9     UNUSED
                  9     2     9
                  11    2     UNUSED
                  13    1     13
                  14    18    UNUSED
                  32    95    32
                  127   1     UNUSED
      BASESET     "ISO Registration Number 100//CHARSET
                  ECMA-94 Right Part of Latin Alphabet Nr. 1//ESC 2/13
      4/1"
      DESCSET     128   32    UNUSED
                  160   95    32
                  255   1     UNUSED

CAPACITY    SGMLREF
            TOTALCAP    150000
            GRPCAP            150000

SCOPE       DOCUMENT
SYNTAX
      SHUNCHAR CONTROLS     0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
18
                        19 20 21 22 23 24 25 26 27 28 29 30 31 127 255
      BASESET     "ISO 646:1983//CHARSET
            International Reference Version (IRV)//ESC 2/5 4/0"
      DESCSET     0     128    0
      FUNCTION    RE          13
                  RS          10
                  SPACE       32
                  TAB SEPCHAR  9
      NAMING            LCNMSTRT ""
                  UCNMSTRT ""
                  LCNMCHAR ".-"
                  UCNMCHAR ".-"
                  NAMECASE    GENERAL YES
                              ENTITY  NO
      DELIM       GENERAL     SGMLREF
                  SHORTREF    SGMLREF
      NAMES       SGMLREF
      QUANTITY    SGMLREF
                  NAMELEN     34
                  TAGLVL            100
                  LITLEN            1024
                  GRPGTCNT    150
                  GRPCNT            64

FEATURES
      MINIMIZE
            DATATAG  NO
            OMITTAG  NO
            RANK     NO
            SHORTTAG NO
      LINK
            SIMPLE   NO
```

Attorney Docket No. 24744-2

```
            IMPLICIT NO
            EXPLICIT NO
      OTHER
            CONCUR      NO
            SUBDOC      NO
            FORMAL      YES
      APPINFO     NONE
>

<!DOCTYPE HTMLPLUS [

<!-- DTD for HTML+

Markup minimisation should be avoided, otherwise the default
<!SGML> declaration is fine.

Browsers should be forgiving of markup errors.

Common Attributes:

   id       the id attribute allows authors to name elements such as
            headers and paragraphs as potential destinations for links.
            Note that links don't specify points, but rather extended
            objects.

   index    allows authors to specify how given headers etc should be
            indexed as primary or secondary keys, where "/" separates
            primary from secondary keys, ";" separates multiple entries
-->

<!-- ENTITY DECLARATIONS

 <!ENTITY % foo "X | Y | Z"> is a macro definition for parameters and in
 subsequent statements, the string "%foo;" is expanded to "X | Y | Z"

 Various classes of SGML text types:

  #CDATA   text which doesn't include markup or entity references

  #RCDATA  text with entity references but no markup

  #PCDATA  text occurring in a context in which markup and entity
           references may occur.
-->

<!ENTITY % URL "CDATA" -- a URL or URN designating a hypertext node -->

<!ENTITY % text "#PCDATA|A|IMG|EM|EMBED|INPUT|BR|CHANGED|NOTE">

<!ENTITY % paras "P|PRE|FIG">

<!ENTITY % lists "UL|OL|DL">

<!ENTITY % misc "HR|TBL|FORM|PANEL|GROUP">

<!ENTITY % heading "H1|H2|H3|H4|H5|H6">

<!ENTITY % table "%text;|P|%heading;|%lists;">

<!ENTITY % main "%heading;|%misc;|%lists;|%paras;|%text;">

<!ENTITY % setup "(TITLE? & HTML? &ISINDEX? & NEXTID? & LINK* & BASE?)">

<!--

<!ELEMENT tagname - - CONTENT> elements needing closing tags

<!ELEMENT tagname - O CONTENT> elements without closing tags

<!ELEMENT tagname - O EMPTY> elements without content or closing tags

   The content definition is:

      a)    an entity definition as defined above

      b)    a tagname
```

```
      c)    (brackets enclosing the above)
   These may be combined with the operators:
      A*    A occurs zero or more times
      A+    A occurs one or more times
      A|B   implies either A or B
      A?    A occurs zero or one times
      A,B   implies first A then B
-->
<!ELEMENT HTMLPLUS O O ((HEAD, BODY) | ((%setup;), (%main;)*))>
<!ELEMENT HEAD - - (%setup;)>
<!ELEMENT BODY - - (%main;)*>
<!-- Document title -->
<!ELEMENT TITLE - - (#PCDATA | EM)+>
<!ATTLIST TITLE
      id       ID      #IMPLIED -- link destination --
      index    CDATA   #IMPLIED -- entries for index compilation -->
<!-- Document role for cataloging documents held by servers -->
<!ELEMENT HTML - O (EMPTY)>
<!ATTLIST HTML role CDATA #IMPLIED -- home page, index, ... -->
<!-- Floating panel which can be moved around relative to the normal text
flow. Often rendered with a different background and possibly framed. The
panel can be anchored to a named point in the document as specified by
the AT attribute. The panel may be placed at that point or after, but not
before.
-->
<!ELEMENT PANEL - - (TITLE?, (%main;)*)>
<!ATTLIST PANEL
      id       ID      #IMPLIED -- defines link destination --
      at       IDREF   #IMPLIED -- anchor point --
      role     CDATA   #IMPLIED -- annotation/aside
      index    CDATA   #IMPLIED -- entries for index compilation -->
<!ELEMENT HR - O EMPTY -- Horizontal Rule -->
<!-- Document headers -->
<!ELEMENT (%heading;) - - (#PCDATA | EM)+>
<!ATTLIST (%heading;)
      id       ID      #IMPLIED -- defines link destination --
      index    CDATA   #IMPLIED -- entries for index compilation -->
<!-- logical emphasis with optional style hints -->
<!ELEMENT EM - - (%text;)*>
<!ATTLIST EM
   role   CDATA  #IMPLIED -- semantic category e.g. CITE --
   b      (b)    #IMPLIED -- render in bold font --
   i      (i)    #IMPLIED -- render in italic font --
   u      (u)    #IMPLIED -- underline text --
   tt     (tt)   #IMPLIED -- render in typewriter font --
   tr     (tr)   #IMPLIED -- render in serif (Times Roman) font --
   hv     (hv)   #IMPLIED -- render in sans serif (Helvetica) font --
   sup    (sup)  #IMPLIED -- superscript --
   sub    (sub)  #IMPLIED -- subscript --
   index  CDATA  #IMPLIED -- entries for index compilation -->
```

```
<!-- Paragraphs with different roles and optional style hints
     Note that paragraphs act as containers for the following text -->

<!ELEMENT P - O (%text;)+>

<!ATTLIST P
   id       ID          #IMPLIED -- link destination --
   role     CDATA       #IMPLIED -- semantic role --
   align    CDATA       #IMPLIED -- left, center or right --
   indent   (indent)    #IMPLIED -- indented margins --
   index    CDATA       #IMPLIED -- entries for index compilation -->

<!ELEMENT BR - O EMPTY -- line break in normal text-->

<!-- Preformatted text with fixed pitch font, respecting original spacing
and newlines. Authors can also request proportional fonts. Further
control is possible with EM, and TAB -->

<!ELEMENT PRE - - (TAB|%text;)+>

<!ATTLIST PRE
   id       ID       #IMPLIED -- link destination --
   role     CDATA    #IMPLIED -- various styles --
   tr       (tr)     #IMPLIED -- serif (Times Roman) font --
   hv       (hv)     #IMPLIED -- sans serif (Helvetica) font --
   width    NUMBER   #IMPLIED -- e.g. 40, 80, 132 --
   index    CDATA    #IMPLIED -- entries for index compilation -->

<!ELEMENT TAB - O EMPTY>

<!ATTLIST TAB
   at  NUMBER  #IMPLIED -- position measured in widths of a capital M --
   align  (left|center|right|decimal)  left -- tab alignment -->

<!-- Lists which can be nested -->

<!ELEMENT OL - - (LI | UL | OL)+ -- ordered list -->

<!ATTLIST OL
   id       ID         #IMPLIED
   compact  (compact)  #IMPLIED
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!ELEMENT UL - - (LI | UL | OL)+ -- unordered list -->

<!ATTLIST UL
   id       ID         #IMPLIED      -- link destination --
   compact  (compact)  #IMPLIED      -- reduced interitem spacing --
   narrow   (narrow)   #IMPLIED      -- narrow perhaps multi columns --
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!-- List items for UL and OL lists -->

<!ELEMENT LI - O (P|%text;)+>

<!ATTLIST LI
   id       ID         #IMPLIED
   src      %URL;      #IMPLIED -- icon for use in place of bullet --
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!-- Definition Lists (terms + definitions) -->

<!ELEMENT DL - - (DT,DD)+ -- DT and DD *MUST* be paired -- > <!ATTLIST DL
   id       ID         #IMPLIED
   compact  (compact)  #IMPLIED
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!ELEMENT DT - O (%text;)+ -- term text -- >

<!ELEMENT DD - O (P|UL|OL|%text;)+ -- definition text -- >

<!ATTLIST (DT|DD)
   id        ID        #IMPLIED
   index     CDATA     #IMPLIED -- entries for index compilation -->
```

```
<!-- Tables with titles and column headers, e.g.

   <tbl border>
      <tt> An Example of a Table
      <th> <th s="2"> average <th> other <tr>
      <th> <th> height <th> weight <th> category <tr>
      <td> males <td> 1.9 <td> .003 <td> yyy <tr>
      <td> females <td> 1.7 <td> .002 <td> xxx
   </tbl>

-->

<!ELEMENT TBL - - (TT?, (TH|TD|TR|TB)*) -- mixed headers and data -->

<!ATTLIST TBL
   id       ID         #IMPLIED
   compact  (compact)  #IMPLIED      -- if present use compact style --
   border   (border)   #IMPLIED      -- if present draw borders --
   index    CDATA      #IMPLIED       -- entries for index compilation -->

<!ELEMENT TT - O (%text;)+ -- table title -->

<!ATTLIST TT top (top) #IMPLIED -- place title above table -->

<!ELEMENT TH - O (%table;)* -- a header cell -->

<!ATTLIST TH
   colspan  NUMBER     1          -- columns spanned --
   rowspan  NUMBER     1          --. rows spanned --
   align    CDATA    #IMPLIED     -- left, center or right -->

<!ELEMENT TD - O (%table;)* -- a data cell -->

<!ATTLIST TD
   colspan  NUMBER     1          -- columns spanned --
   rowspan  NUMBER     1          --. rows spanned --
   align    CDATA    #IMPLIED     -- left, center or right -->

<!ELEMENT TR - O EMPTY -- row separator -->

<!ELEMENT TB - O EMPTY -- vertical break of 1/2 line spacing -->

<!-- Forms composed from input fields and selection menus

These elements define fields which users can type into or select with
mouse clicks. The browser should manage the input focus e.g. with the
tab/shift tab keys and mouse clicks.

The enter/return key is then taken to mean the use has filled in the form
and wants the apppropriate action taken:
   - send as query/update to WWW server
   - email/fax to designated person

The action is specified as a URL, e.g. "mailto:dsr@hplb.hpl.hp.com You
can specify additional mail headers with the MH tag:

<MH>Subject: Please add me to tennis tournament</MH>

Each FORM should include one or more INPUT elements which can be layed
out with normal and preformatted text, lists and tables.
 -->

<!ELEMENT FORM - - (MH, (%main;)*)>

<!ATTLIST FORM
   id       ID         #IMPLIED
   action   %URL;      #IMPLIED
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!ELEMENT MH - - CDATA -- one or more RFC 822 header fields -->

<!ATTLIST MH hidden (hidden) #IMPLIED -- hide the mail headers from view -->
```

```
<!-- INPUT elements should be defined within a FORM element.

Users can alter the value of the INPUT element by typing or clicking with
the mouse. Use radio buttons for selecting one attribute value from a set
of alternatives. In this case there will be several INPUT elements with
the same name. Attributes which can take multiple values at the same time
should be defined with checkboxes: define each allowed value in a
separate INPUT element but with the same attribute name. For checkboxes
and radio buttons, the value doesn't change, instead the state of the
button shown by the presence or absence of the checked attribute in each
element.

The size attribute specifies the size of the input field as appropriate
to each type. For text this gives the width in characters and height in
lines (separated by an "x"). For numbers this gives the maximum
precision.

-->

<!ELEMENT INPUT - O EMPTY>

<!ATTLIST INPUT
   name     CDATA     #IMPLIED -- attribute name (may not be unique) --
   type     CDATA     #IMPLIED --TEXT,URL,INT,FLOAT,DATE,CHECKBOX,RADIO--
   size     CDATA     #IMPLIED -- e.g."32x4" for multiline text --
   value    CDATA     #IMPLIED -- attribute value (altered by user) --
   checked  (checked) #IMPLIED -- for check boxes and radio buttons --
   disabled(disabled),  #IMPLIED    -- if grayed out --
   error    (error)   #IMPLIED -- if in error -->

<!-- Embedded Data

You can embed information in a foreign format into the HTML+ document.
This is very convenient for mathematical equations and simple drawings.
Images and complex drawings are better specified as linked documents
using the FIG or IMG elements.

Arbitrary 8 bit data is allowed but any occurrences of the following
chars must be escaped as shown:

      "&"     by      "&amp;"
      "<"     by      "&lt;"
      ">"     by      "&gt;"
The browser can pipe such data thru filters to generate the corresponding
pixmap The data format is specified as a MIME content type, e.g.
"text/eqn"

It would have been nice to use the NOTATION type in place of CDATA, but
this doesn't
appear to be practical with externally specified content type names.
-->

<!ELEMENT EMBED - - (RCDATA)>
<!ATTLIST EMBED
   id       ID        #IMPLIED
   type     CDATA     #IMPLIED  -- mime content type --
   index    CDATA     #IMPLIED -- entries for index compilation -->

<!-- Figures

The image/drawing is specified by a URL or as embedded data for simple
drawings. The element's text serves as the caption. Use the emphasis with
style = "credits" to record photo credits etc.

-->
```

```
<!ELEMENT FIG - - (EMBED?, FIGD?, (FIGA|FIGT)*, (%text;)*)>

<!ATTLIST FIG
   id       ID         #IMPLIED
   align    CDATA      #IMPLIED -- position: left, right or center --
   cap      CDATA      #IMPLIED -- caption at left, right, top, bottom --
   noflow   (noflow)   #IMPLIED -- disables text flow --
   ismap    (ismap)    #IMPLIED -- server can handle mouse clicks/drags --
   src      %URL;      #IMPLIED -- link to image data --
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!ELEMENT FIGD - - (%table;) -- figure description -->

<!-- Figure anchors designate polygonal areas on the figure which can be
clicked with the mouse. The default area is the whole of the figure. This
mechanism interprets mouse clicks locally, and browsers can choose to
highlight the designated area (or change the mouse sprite) when the mouse
is moved over the area.

Note that polygons may be non-convex or even intersect themselves,
thereby complicating the definition of what is enclosed by the polygon.
Holes are excluded.
-->

<!ELEMENT FIGA - O EMPTY>

<!ATTLIST FIGA
   href     %URL;     #REQUIRED -- link to traverse when clicked --
   area     NUMBERS   #IMPLIED -- x1,y1,x2,y2,x3,y3,... -->

<!-- FIGT Text on top of an figure background, or in a colored background
box which sits arbitrarily on top of an figure background. The text can
include headers, lists and tables etc. The width attribute allows you to
limit the width of the text box. The height is then determined
automatically by the browser.

FIGT can also be used to position a graphic on top of a picture using an
IMG element within FIGT. In this case the chromakey attribute may allow
parts of the underlying image to show through.

You can make the whole of the box into a hypertext link. This will act as
if it is underneath any hypertext links specified by the overlay markup
itself.

FIGT can also be used for annotations on figures, and held separately
from the document with the figure to which they apply. In this case, use
the idref attribute to name which figure is appropriate.
-->

<!ELEMENT FIGT - - (%main;)>

<!ATTLIST FIGT
   idref  IDREF    #IMPLIED -- names FIG element if held separately --
   at     NUMBERS  #IMPLIED -- upper left origin for text --
   width  NUMBER   #IMPLIED -- given as fraction of picture --
   framed (framed) #IMPLIED -- framed with coloured background --
   href   %URL;    #IMPLIED -- link to traverse when clicked -->

<!-- inline icons/small graphics
The align attribute defines whether the top middle or bottom of the
graphic and current text line should be aligned vertically

The SEETHRU attribute is intended as a chromakey to allow a given colour
to be designated as "transparent". Pixels with this value should not be
painted. The exact format of this attribute's value has yet to be
defined.


Use the FIG tag for captioned figures with active areas etc.
-->

<!ELEMENT IMG - O EMPTY>
```

```
<!ATTLIST IMG
   src     %URL;   #REQUIRED -- where to get image data --
   align   CDATA   #IMPLIED  -- top, middle or bottom --
   seethru CDATA   #IMPLIED  -- for transparency --
   ismap   (ismap) #IMPLIED  -- send mouse clicks/drags to server -->

<!-- Hierarchical groups for books, chapters, sections etc. -->

<!ELEMENT GROUP - - ((TITLE|LINK*), (%main;)*)>

<!ATTLIST GROUP
   id    ID        #IMPLIED
   role  CDATA     #IMPLIED -- book, chapter, section etc. --
   inset (inset)   #IMPLIED -- rendering hint: indent margins -->

<!-- change bars defined by a matched pair of CHANGED elements:

      <changed id=z34> changed text <changed idref=z34>

This tag can't act as a container, since changes don't respect
the nesting implied by paragraphs, headers, lists etc.
-->

<!ELEMENT CHANGED - O EMPTY>

<!ATTLIST CHANGED -- one of id and idref is always required --
   id       ID      #IMPLIED -- signals start of changes --
   idref    IDREF   #IMPLIED -- signals end of changes -->

<!-- Matched pairs of NOTE elements can be used to delimit text
associated with one or more annotations. The annotation itself can be
linked via an explicit URL or defined as one or more PANEL elements
naming this NOTE.

      <note id=z34> delimited text <note idref=z34>

This tag can't act as a container, since users may select text without
regard to
the nesting implied by paragraphs, headers, lists etc.
-->

<!ELEMENT NOTE - O EMPTY>

<!ATTLIST NOTE -- one of id and idref is always required --
   id       ID      #IMPLIED -- signals start of delimited text --
   href     %URL;   #IMPLIED -- for directly specifying the annotation --
   idref    IDREF   #IMPLIED -- signals end of delimited text -->

<!-- Hypertext Links from points within document nodes -->

<!ELEMENT A - - (#PCDATA | IMG | EM | EMBED)*>

<!ATTLIST A
   id       ID      #IMPLIED -- as target of link --
   name     CDATA   #IMPLIED -- for backwards compatibility with HTML--
   href     %URL;   #IMPLIED -- destination node --
   role     CDATA   #IMPLIED -- role of link, e.g. "partof" --
   effect   CDATA   #IMPLIED -- replace/new/overlay --
   print    CDATA   #IMPLIED -- reference/footnote/section --
   title    CDATA   #IMPLIED -- when otherwise unavailable --
   type     CDATA   #IMPLIED -- for presentation cues --
   size     NAMES   #IMPLIED -- for progress cues -->
```

```
<!-- Other kinds of relationships between documents -->

<!ELEMENT LINK - O EMPTY>

<!ATTLIST LINK
   href     %URL;     #IMPLIED -- destination node --
   role     CDATA     #IMPLIED -- role played, e.g. "toc" -->

<!-- Original document URL for resolving relative URLs  -->

<!ELEMENT BASE - O EMPTY>
<!ATTLIST BASE HREF %URL; #IMPLIED>

<!-- Signifies the document's URL accepts queries -->

<!ELEMENT ISINDEX - O (EMPTY)>
<!ATTLIST ISINDEX href %URL; #IMPLIED -- defaults to document's URL -->

<!-- For use with autonumbering editors - don't reuse ids, allocate next
one starting from this one -->

<!ELEMENT NEXTID - O (EMPTY)>
<!ATTLIST NEXTID N NAME #REQUIRED>

<!-- Mnemonic character entities for 8 bit ANSI Latin-1 -->

<!ENTITY iexcl "&#161;" -- inverted exclamation mark -->
<!ENTITY cent "&#161;" -- cent sign -->
<!ENTITY pound "&#163;" -- pound sign -->
<!ENTITY yen "&#165;" -- yen sign -->
<!ENTITY brvbar "&#166;" -- broken vertical bar -->
<!ENTITY sect "&#167;" -- section sign -->
<!ENTITY copy "&#169;" -- copyright sign -->
<!ENTITY laquo "&#171;" -- angle quotation mark, left -->
<!ENTITY raquo "&#187;" -- angle quotation mark, right -->
<!ENTITY not "&#172;" -- negation sign -->
<!ENTITY reg "&#174;" -- circled R registered sign -->
<!ENTITY deg "&#176;" -- degree sign -->
<!ENTITY plusmn "&#177;" -- plus or minus sign -->
<!ENTITY sup2 "&#178;" -- superscript 2 -->
<!ENTITY sup3 "&#179;" -- superscript 3 -->
<!ENTITY micro "&#181;" -- micro sign -->
<!ENTITY para "&#182;" -- paragraph sign -->
<!ENTITY sup1 "&#185;" -- superscript 1 -->
<!ENTITY middot "&#183;" -- center dot -->
<!ENTITY frac14 "&#188;" -- fraction 1/4 -->
<!ENTITY frac12 "&#189;" -- fraction 1/2 -->
<!ENTITY iquest "&#191;" -- inverted question mark -->
<!ENTITY frac34 "&#190;" -- fraction 3/4 -->
<!ENTITY AElig "&#198;" -- capital AE diphthong (ligature) -->
<!ENTITY Aacute "&#193;" -- capital A, acute accent -->
<!ENTITY Acirc "&#194;" -- capital A, circumflex accent -->
<!ENTITY Agrave "&#192;" -- capital A, grave accent -->
<!ENTITY Aring "&#197;" -- capital A, ring -->
<!ENTITY Atilde "&#195;" -- capital A, tilde -->
<!ENTITY Auml "&#196;" -- capital A, dieresis or umlaut mark -->
<!ENTITY Ccedil "&#199;" -- capital C, cedilla -->
<!ENTITY ETH "&#208;" -- capital Eth, Icelandic -->
<!ENTITY Eacute "&#201;" -- capital E, acute accent -->
<!ENTITY Ecirc "&#202;" -- capital E, circumflex accent -->
<!ENTITY Egrave "&#200;" -- capital E, grave accent -->
<!ENTITY Euml "&#203;" -- capital E, dieresis or umlaut mark -->
<!ENTITY Iacute "&#205;" -- capital I, acute accent -->
<!ENTITY Icirc "&#206;" -- capital I, circumflex accent -->
<!ENTITY Igrave "&#204;" -- capital I, grave accent -->
<!ENTITY Iuml "&#207;" -- capital I, dieresis or umlaut mark -->
<!ENTITY Ntilde "&#209;" -- capital N, tilde -->
<!ENTITY Oacute "&#211;" -- capital O, acute accent -->
<!ENTITY Ocirc "&#212;" -- capital O, circumflex accent -->
<!ENTITY Ograve "&#210;" -- capital O, grave accent -->
```

```
<!ENTITY Oslash "&#216;" -- capital O, slash -->
<!ENTITY Otilde "&#213;" -- capital O, tilde -->
<!ENTITY Ouml "&#214;" -- capital O, dieresis or umlaut mark -->
<!ENTITY THORN "&#222;" -- capital THORN, Icelandic -->
<!ENTITY Uacute "&#218;" -- capital U, acute accent -->
<!ENTITY Ucirc "&#219;" -- capital U, circumflex accent -->
<!ENTITY Ugrave "&#217;" -- capital U, grave accent -->
<!ENTITY Uuml "&#220;" -- capital U, dieresis or umlaut mark -->
<!ENTITY Yacute "&#221;" -- capital Y, acute accent -->
<!ENTITY aacute "&#225;" -- small a, acute accent -->
<!ENTITY acirc "&#226;" -- small a, circumflex accent -->
<!ENTITY aelig "&#230;" -- small ae diphthong (ligature) -->
<!ENTITY agrave "&#224;" -- small a, grave accent -->
<!ENTITY amp "&amp;" -- ampersand -->
<!ENTITY aring "&#229;" -- small a, ring -->
<!ENTITY atilde "&#227;" -- small a, tilde -->
<!ENTITY auml "&#228;" -- small a, dieresis or umlaut mark -->
<!ENTITY ccedil "&#231;" -- small c, cedilla -->
<!ENTITY eacute "&#233;" -- small e, acute accent -->
<!ENTITY ecirc "&#234;" -- small e, circumflex accent -->
<!ENTITY egrave "&#232;" -- small e, grave accent -->
<!ENTITY eth "&#240;" -- small eth, Icelandic -->
<!ENTITY euml "&#235;" -- small e, dieresis or umlaut mark -->
<!ENTITY gt "&#62;" -- greater than -->
<!ENTITY iacute "&#237;" -- small i, acute accent -->
<!ENTITY icirc "&#238;" -- small i, circumflex accent -->
<!ENTITY igrave "&#236;" -- small i, grave accent -->
<!ENTITY iuml "&#239;" -- small i, dieresis or umlaut mark -->
<!ENTITY lt "&lt;" -- less than -->
<!ENTITY ntilde "&#241;" -- small n, tilde -->
<!ENTITY oacute "&#243;" -- small o, acute accent -->
<!ENTITY ocirc "&#244;" -- small o, circumflex accent -->
<!ENTITY ograve "&#242;" -- small o, grave accent -->
<!ENTITY oslash "&#248;" -- small o, slash -->
<!ENTITY otilde "&#245;" -- small o, tilde -->
<!ENTITY ouml "&#246;" -- small o, dieresis or umlaut mark -->
<!ENTITY szlig "&#223;" -- small sharp s, German (sz ligature) -->
<!ENTITY thorn "&#254;" -- small thorn, Icelandic -->
<!ENTITY uacute "&#250;" -- small u, acute accent -->
<!ENTITY ucirc "&#251;" -- small u, circumflex accent -->
<!ENTITY ugrave "&#249;" -- small u, grave accent -->
<!ENTITY uuml "&#252;" -- small u, dieresis or umlaut mark -->
<!ENTITY yacute "&#253;" -- small y, acute accent -->
<!ENTITY yuml "&#255;" -- small y, dieresis or umlaut mark -->


<!-- other entities -->

<!ENTITY ndash "--"  -- En dash -->
<!ENTITY mdash "---" -- Em dash -->

<!ENTITY nbsb " " -- non breaking space -->
<!ENTITY ensp "  " -- En space -->
<!ENTITY emsp "   " -- Em space -->

<!ENTITY shy "-" -- soft hyphen -->


<!-- The END -->
]>
```

PE JC64
OCT 2 4 2003
PATENT & TRADEMARK OFFICE

Appendix II - Entity Definitions

The following character definitions conform to widely available 8 bit character sets, e.g. the ANSI Latin-1 character set which is available for both the PC and X11. The corresponding 8-bit character codes are given in the DTD. All definitions conform to the ISO 8879-1986 naming conventions.

Æ	capital AE diphthong (ligature)
Á	capital A, acute accent
Â	capital A, circumflex accent
À	capital A, grave accent
Å	capital A, ring
Ã	capital A, tilde
Ä	capital A, dieresis or umlaut mark
Ç	capital C, cedilla
Ð	capital Eth, Icelandic
É	capital E, acute accent
Ê	capital E, circumflex accent
È	capital E, grave accent
Ë	capital E, dieresis or umlaut mark
Í	capital I, acute accent
Î	capital I, circumflex accent
Ì	capital I, grave accent
Ï	capital I, dieresis or umlaut mark
Ñ	capital N, tilde
Ó	capital O, acute accent
Ô	capital O, circumflex accent
Ò	capital O, grave accent
Ø	capital O, slash
Õ	capital O, tilde
Ö	capital O, dieresis or umlaut mark
Þ	capital THORN, Icelandic
Ú	capital U, acute accent
Û	capital U, circumflex accent
Ù	capital U, grave accent
Ü	capital U, dieresis or umlaut mark
Ý	capital Y, acute accent
á	small a, acute accent
â	small a, circumflex accent
æ	small ae diphthong (ligature)
à	small a, grave accent
å	small a, ring
ã	small a, tilde
ä	small a, dieresis or umlaut mark
ç	small c, cedilla
é	small e, acute accent
ê	small e, circumflex accent
è	small e, grave accent
ð	small eth, Icelandic

10/24/03 3:47 PM

ë	small e, dieresis or umlaut mark
í	small i, acute accent
î	small i, circumflex accent
ì	small i, grave accent
ï	small i, dieresis or umlaut mark
ñ	small n, tilde
ó	small o, acute accent
ô	small o, circumflex accent
ò	small o, grave accent
ø	small o, slash
õ	small o, tilde
ö	small o, dieresis or umlaut mark
ß	small sharp s, German (sz ligature)
þ	small thorn, Icelandic
ú	small u, acute accent
û	small u, circumflex accent
ù	small u, grave accent
ü	small u, dieresis or umlaut mark
ý	small y, acute accent
ÿ	small y, dieresis or umlaut mark

In addition, there are some common publishing characters:

¡	inverted exclamation mark
¢	cent sign
£	pound sign
¥	yen sign
¦	broken vertical bar
§	section sign
©	copyright sign
«	angle quotation mark, left
»	angle quotation mark, right
¬	negation sign
®	circled R registered sign
°	degree sign
±	plus or minus sign
²	superscript 2
³	superscript 3
µ	micro sign
¶	paragraph sign
¹	superscript 1
·	center dot
¼	fraction 1/4
½	fraction 1/2
¾	fraction 3/4
¿	inverted question mark

Finally, there are several special purpose definitions:

–	En sized horizontal dash (--)
—	Em sized horizontal dash (---)
	Non-breaking space
	En space ( )
	Em space ( )
	Soft hyphen

OIPE JC64
OCT 2 4 2003
PATENT & TRADEMARK OFFICE

Appendix III - Compatibility with HTML

HTML+ browsers should be able to view HTML documents with very little extra code and it is strongly recommended that browsers support both formats. Older HTML browsers will be able to view HTML+ documents which don't contain figures, tables or forms.

Lists

HTML	HTML+
<menu>	<ul compact>
<dir>	<ul narrow>

Emphasis

HTML+ replaces the various tags used by HTML with a single tag. It may be worth changing the name for the emphasis tag in HTML+ from EM to EM, to gain compatibility with this common form. However, using EM might be confused with the typographical term *em* as in em dash (you also get en dash). EM has the merit of being unambigous.

HTML	HTML+
<tt>	<em tt>
	<em b>
<b>	<em b>
<i>	<em i>
<u>	<em u>
<code>	
<samp>	
<kbd>	
<var>	
<dfn>	
<cite>	

Miscellaneous

Some tags which are deprecated in HTML are now obsolete, and should be mapped to preformatted text. This doesn't work quite right as PRE assumes that characters such as "<", ">" and "&" have been replaced by their entity definitions. Browsers should perhaps treat "<" as verbatim unless it forms part of an expected tag. In this way, unescaped occurrences of these three characters will normally display as intended.

HTML	HTML+
<plaintext>	<pre>
<xmp>	<pre>
<listing>	<pre>

The following two tags have been absorbed into the standard mechanism for paragraphs:

<address>	becomes	<p role="byline" align="right">
<blockquote>	becomes	<p role="quote">

Notes for Implementors

Please ensure that browsers can tolerate bad markup. In practice, this is quite straightforward to achieve, provided a naive top-down SGML parser is avoided. A forgiving parser should be able to cope with tags in unexpected positions, e.g. the <A> tag bracketing a header[16]. Unknown tags should be simply ignored.

Implementors should endevour to make sure that documents can be scrolled efficiently regardless of their length. Always parsing from the start of the document leads to jerky performance. Two strategies for efficiently scrolling through documents are:

a) Establish regular landmarks throughout the document for which the state of the parse is known. The browser can then work forward from the nearest landmark, when it needs to refresh the screen after a scroll operation. The landmarks need updating when users make changes, while using a WYSIWYG editor.

b) When scrolling up, parse backwards to work out the state at earlier points in the document. This can be done via a combination of skipping back, looking for markup which causes a line break etc. and then parsing forward until the current position, to find the change of state. This can be repeated until the parser reaches a point prior to the new top of the window.

Practical experience has shown the importance of providing cues to users on progress in retrieving documents over the network. These will depend on the protocol, but should show at least how much data has been received at any point. The network connections shouldn't block, and an abort button is essential[17]. It is generally better to avoid displaying the retrieved document in a new window, unless explicitly requested by the user, e.g. by holding down the shift key when clicking the hypertext link.

References

This is missing the appropriate references to work on the syntax and name service for URNs. The HTTP definition needs updating to cover the encoding of form data (and *ismap* ?).

[Berners-Lee 93a] "*Hypertext Markup Language (HTML)*", Tim Berners-Lee, March 1993. URL=`ftp://info.cern.ch/pub/www/doc/http-spec.ps`

[Berners-Lee 93b] "*Uniform Resource Locators*", Tim Berners-Lee, January 1992. URL=`ftp://info.cern.ch/pub/ietf/url4.ps`

[Berners-Lee 93c] "*Protocol for the Retrieval and Manipulation of Textual and Hypermedia Information*", Tim Berners-Lee, 1993. URL=`ftp://info.cern.ch/pub/www/doc/html-spec.ps`

[Goldfarb 90] "*The SGML Handbook*", Charles F. Goldfarb, Clarendon Press · Oxford.

[Kimber 93] Article in comp.text.sgml newsgroup, 24th May 1993 by Elliot Kimber (drmacro@vnet.almaden.ibm.com), URL=`news:19930524.152345.29@almaden.ibm.com`

[Raisch 93] "*Style sheets for HTML*", Robert Raisch, June 1993, O'Reilly & Associates email: `raisch.ora.com`

---

[16]Headers typically cause a line break and leave a vertical gap. If the hypertext link definition is parsed prior to the beginning of the header, the starting position for the button will be in the wrong place - browsers should therefore adjust this position to the beginning of the text.

[17]For X11 on Unix systems, the *select* system call can be used with non-blocking I/O to poll the event queue at regular intervals. The *XtAddInput* call acts as a wrapper around *select* for this very purpose. Users can then continue to view the current document as well as being able to click an abort button (which sets a global variable, polled by the comms software). Be careful to disable unsafe actions, e.g. trying to get a second document while still waiting to get the first (a race hazard).

h02r20" Emhm2E80

OIPE
OCT 2 4 2003
PATENT & TRADEMARK OFFICE

HTML+ support for eqn & Postscript

Dave_Raggett <dsr@hplb.hpl.hp.com>

- **Mail folder:** WWW Talk Apr-Jun 1993 Archives
- **Next message:** Marc Andreessen: "Re: SPaces and Tabs in HTML documents"
- **Previous message:** Tim Berners-Lee: "Re: SPaces and Tabs in HTML documents"
- **Reply:** Torben Noerup Nielsen: "re: HTML+ support for eqn & Postscript"
- **Reply:** Bill Janssen: "Re: HTML+ support for eqn & Postscript"
- **Reply:** Bill Janssen: "Re: HTML+ support for eqn & Postscript"

From: Dave_Raggett <dsr@hplb.hpl.hp.com>
Message-id: <9306140936.AA00563@manuel.hpl.hp.com>
Subject: HTML+ support for eqn & Postscript
To: torben@hawaii.edu, janssen@parc.xerox.com
Date: Mon, 14 Jun 93 10:36:40 BST
Cc: www-talk@nxoc01.cern.ch
Mailer: Elm [revision: 66.36.1.1]

Torben Nielsen says:

> I realize this has come up before, but how about really doing something about
> equation support? There are lots of documents I would like to put into the
> Web, but without support for embedded equations, it's really quite difficult.
> Something simple like eqn support would be great. And eqn shouldn't be all
> that hard to parse either.....

Bill Janssen chips in with:

> And I really like to send encapsulated Postcript in my documents...
> Having ghostscript should make the parsing and layout easy!

Well both of these will be possible with the HTML+ DTD, by using the capability to embed foreign formats inline in the HTML+ source, e.g.

```
    <H2>A example of an equation</H2>

    <EMBED TYPE="text/eqn">zeta (s) ~=~ sum from k=1 to inf
     k sup -s ~~~ (Re s &gt; 1) </EMBED>
```

The browser identifies the format of the embedded data from the "type" attribute, specified as a MIME content type. Certain characters need to be escaped using entity definitions, e.g. ">" by ">" in the example.

Building in support for a range of formats has the danger of leading to very large programs for browsers. This could be avoided by using a common API for rendering foreign formats, e.g. as functions that take a sequence of bytes and return a pixmap.

Browsers can then be upgraded to display new formats without changing their code at all. All you would need is a way of binding the MIME content type to the function name for that format, e.g. via X resources or a config file. The functions could be implemented as separate programs driven via pipes and stdin/stdout or as dynamically linked library modules (Windows DLLs).

How does that sound?

Dave

p.s. you can also put the foreign data in a separate file referenced by a URL.

Document Retrieved

Exhibit C

RECEIVED
OCT 2 9 2003
OFFICE OF PETITIONS

OIPE JC64
OCT 24 2003
PATENT & TRADEMARK OFFICE

IPE JC
OCT 2 4 2003
PATENT & TRADEMARK OFFICE

WWW Workshop Announcement

Dave_Raggett <dsr@hplb.hpl.hp.com>

- **Mail folder:** WWW Talk Jul-Oct 1993
- **Next message:** Brian Smithson: "Re: Search keyword dialog box in Mosaic 2.0pre0?"
- **Previous message:** David Martland: "Mail messages in Mosaic"


```
From: Dave_Raggett <dsr@hplb.hpl.hp.com>
Message-id: <9307231606.AA26875@manuel.hpl.hp.com>
Subject: WWW Workshop Announcement
To: www-talk@nxoc01.cern.ch
Date: Fri, 23 Jul 93 17:06:18 BST
Mailer: Elm [revision: 66.36.1.1]
Status: RO

For those of you off to Cambridge for the workshop, there is a revised
version (dated 23rd July) of the HTML+ spec at:

        ftp://15.254.100.100/pub/htmlplus.ps

An older version can be found at:

        http://info.cern.ch/hypertext/WWW/MarkUp/htmlplus.ps

Can I suggest that you try and print off a copy to take with you.
Unfortunately I won't be able to come myself, but hope that there will
be discussion of the spec and look forward to finding out how you all
get on.

I am hoping to get the HTML+ spec accepted as an Internet Draft as soon as
practical, and would very much like help with checking the spec for technical
feasibility and improving clarity etc.

Have fun,

Dave Raggett

p.s. There seems to be a Ctrl-D at the start of the file, an annoying
artifact generated by Microsoft Word! This shouldn't cause problems though.
```

Exhibit D

RECEIVED
OCT 2 9 2003
OFFICE OF PETITIONS

PATENT & TRADEMARK OFFICE
OCT 2 4 2003

OCT 2 4 2003
PATENT & TRADEMARK OFFICE

Re: Lets keep the web together

marca@ncsa.uiuc.edu (Marc Andreessen)

- **Mail folder:** WWW Talk 1992 Archives
- **Next message:** Robert Raisch: "Re: Lets keep the web together "
- **Previous message:** Dan Connolly: "Re: Lets keep the web together "
- **In-reply-to:** Dan Connolly: "Re: Lets keep the web together "
- **References:** Tim Berners-Lee: "Lets keep the web together"


```
Date: Tue, 1 Dec 92 12:15:42 -0800
From: marca@ncsa.uiuc.edu (Marc Andreessen)
Message-id: <9212012015.AA22660@wintermute.ncsa.uiuc.edu>
To: Dan Connolly <connolly@pixel.convex.com>
Cc: timbl@nxoc01.cern.ch, www-talk@nxoc01.cern.ch
Subject: Re: Lets keep the web together
In-reply-to: <9212011804.AA24775@pixel.convex.com>
References: <9212011552.AA01907@www3.cern.ch>
        <9212011804.AA24775@pixel.convex.com>

Dan Connolly writes:
> You can't use any of the parts without using the whole thing: you
> can't use the parser unless you use the HText data structures,
> including the HTMainText global variable! [...]

I have to agree with this as a negative assessment.  Trying to figure
out how to deal with the library hasn't been very much fun.

> Again, we see the need for a CSCW platform.

Well, for what it's worth, the server running on
hoohoo.ncsa.uiuc.edu:80 will accept the ANNOTATE command as proposed
in a previous message of mine, and I can send the patches to enable
this (as well as modify the linemode browser and library to provide
annotation functionality) to anyone who wants them.  A more complex
system is certainly desirable, though.

> I think that if we archived the www-talk mailing list, and built an
> HTTP server that would serve the articles up with hypertext links
> between folloups etc, it would be a good start.
>
> If I code up a server in perl that can do this, will somebody
> provide disk space and network access?

I would, but it already exists:

        http://eies2.njit.edu:80/wmail.html

Dan, if you want to do a mailing list archive also, contact me and
we'll set it up...

Marc
```

OIPE JC64
OCT 2 4 2003
PATENT & TRADEMARK OFFICE

10/24/2003 DHAGOS 00000020 10020030

COPY OF PRIOR ART FOR PATENT OWNER

Exhibit A

ALL-STATE LEGAL™ 800-222-0510

OIPE JC64
OCT 2 4 2003
PATENT & TRADEMARK OFFICE

Request for Comments: XXXX David Raggett (Hewlett Packard)
23rd July 1993

HTML+ (Hypertext markup language)

A proposed standard for a light weight presentation independent delivery format for browsing and querying information across the Internet

Status of this Memo

This document is a proposal for an Internet Draft, and specifies the HTML+ wide-area hypertext document format, with a view to requesting discussion[1] and suggestions for improvements. Distribution of this memo is unlimited.

Abstract

HTML+ is a simple SGML based format for wide-area hypertext documents, for use within the World Wide Web. Unlike desktop publishing formats, HTML+ captures the logical intent of authors. This simplifies the task of writing documents, and permits them to be effectively rendered on a wide range of display types as well as the printed page.

HTML+ represents a substantial improvement over the existing format: HTML, offering nested lists, figures, embedded data in foreign formats for equations etc, tables with support for titles and column headings, change bars, entry forms for querying and updating information sources and for use as questionaires for mailing. This document specifies the HTML+ format with guidelines on how it should be rendered by browsers.

Application Number 10/690,402
Filing Date October 21, 2003
Inventor Dave Raggett
Attorney Docket Number 10030199-1

Introduction

The World Wide Web is a wide area client-server architecture for retrieving hypermedia documents across the Internet. It also supports a means for searching remote information sources, for example bibliographies, phone directories and instruction manuals. There are three main ingredients:

a) Universal naming scheme for documents. The universal resource location syntax specifies documents in terms of the protocol to be used to retrieve them, their Internet host and path name. A format for location independent lifetime identifiers is currently being defined by working groups of the IETF. A network protocol will allow universal resource numbers (URNs) to be resolved to the URL for the nearest available copy.

b) Use of available protocols for retrieving documents over the network, including FTP, NNTP, WAIS, Gopher, and HTTP. The latter is designed specifically for use with the World Wide Web, and combines efficiency with an ability to flexibly exchange information between clients and servers.

c) A document format supporting hypertext links based on URLs and URNs which can specify documents anywhere in the Internet. HTML+ is designed for rendering on a wide variety of different display types and platforms.

Information browsers can display information in a wide variety of formats, e.g. plain text, rich text in the HTML+ format, images in the GIF and JPEG formats, MPEG movies, and MIME documents. The hypertext format has a special significance as it allows users to navigate from one document to the next at the click of a button. It provides the basis for menus, cross references, either within a document or to other documents,

---

[1]Please mail comments to the author dsr@hplb.hpl.hp.com, or to the WWW discussion group: www-talk@nxoc01.cern.ch

OIPE JC64
OCT 2 4 2003
PATENT & TRADEMARK OFFICE

perhaps on the other side of the world. It also provides a means of building larger scale collections of documents that act as journals, books or encyclopedias. The format is also intended to act as a building block for creating wide area groupware applications.

HTML+ follows on from an earlier standard - HTML; see [Berners-Lee 93a], which has been widely used as the basis for hypertext documents in the World Wide Web. The new format grew out of experience with HTML, culminating in the desire to add new features, e.g. inline images, tables, and form fields for greater flexibility in querying remote information sources. This document specifies the HTML+ format and suggests ways in which browsers can choose to render it on a variety of different display types.

2. HTML+ and SGML

HTML+ itself is based on the Standard General Markup Language (SGML), an international standard for document markup that is becoming increasingly important. The term markup derives from the way proof-readers have traditionally pencilled in marks that indicate how the document should be revised.

SGML grew out of a decade of work addressing the need for capturing the logical elements of documents as opposed to the processing functions to be performed on those elements. SGML is essentially an extensible document description language, based on a notation for embedding tags into the body of a document's text. It is defined by the international standard ISO 8879. The markup structure permitted for each class of documents is defined by an SGML Document Type Definition, usually abbreviated to DTD.

Working groups in ISO have recently produced a range of SGML DTDs for documents, e.g. ISO 12083 defines DTDs for books and ISO 10744, which defines the HyTime standard for hypermedia/time-based documents. These standards are large and complex, and perhaps best suited as interchange standards that facilitate conversion between proprietary document formats. By contrast, HTML+ provides a lightweight delivery format that can be rendered by relatively simple browsers, and which has grown out of two years practical experience with wide-area hypertext information systems in the Internet community.

HTML+ and HyTime

The HyTime standard provides a rich range of architectural forms, but is not aimed at run-time efficiency. Suggestions have been made as to how the HTML DTD could be adapted to comply with HyTime's clink architectural form [Kimber 93]. This would necessitate documents declaring links as external entities and the use of local names in link definitions, but in the absence of any immediate benefit, there has been little enthusiasm for this within the World Wide Web community. Instead, it is believed that a straightforward filter program should be used to map HTML and HTML+ documents into a format which is strictly compliant with HyTime, when this becomes appropriate.

A simple example of HTML+

The following is a simple example of an HTML+ document,which illustrates the basic ideas involved in SGML.

```
<title>A Simple HTML+ Document</title>

<h1 id="a1">This is a level one header</h1>

<p> This is some normal text which will wrap at the window margin. You
can emphasise <em>parts of the text</em> if you wish. </p>

<p> This is a new paragraph. Notice that unlike title and header tags,
the matching end tag is optional.
```

The text of the document includes tags which are enclosed in <angle brackets>. Many tags have matching end tags for which the tag name is preceded by the "/" character. The tags are used to markup the document's logical elements, for example, the title, headers and paragraphs. Tags may also be accompanied by parameters, e.g. the "id" attribute in the header tag, which is used to define potential destinations for hypertext jumps.

Unlike most document formats, SGML leaves out the processing instructions that determine the precise appearence of the document, for example the font name and point size, the margins, tab settings and how much white space to leave before and after different elements. The rendering software makes these choices for itself (perhaps guided by user preferences), and so can avoid problems with different page sizes or missing fonts.

Logical markup also preserves essential distinctions that are often lost by lower level procedural formats, making it easier to carry out operations like indexing, and conversion into other document formats.

Practical experience has shown that people often make mistakes when they have to type in the markup for themselves. As a result, most browsers are tolerant of bad markup. This problem is being minimised by keeping the format as simple as possible and encouraging the development of WYSIWYG editors.

The HTML+ Document Format

The following sections go through the various features of the format with suggestions as to how browsers should render them. The DTD for HTML+ is given in Appendix I.

Parsing HTML+ Documents

By default, HTML+ documents are made up of 8-bit characters in the ISO 8859 Latin-1 character set. In future, 16 bit character sets may be used to cover a wider range of languages. The HTTP network protocol uses the MIME standard (RFC 1341) to specify the document type and the character set. It is assumed that the chosen character set includes the printable 7 bit US ASCII characters as a subset.

The DTD specifies the syntax of the document structure, in particular, which tags are permitted in any given context. Certain tags are only permitted at the start of the document. Tags and attribute names are case insensitive, thus <TITLE> is equivalent to <title>. Minimisation is forbidden to avoid problems with parsing unknown tags.

In general, SGML entity definitions are used to represent characters which would otherwise be confused with markup elements:

&	is represented by	`&`
<	is represented by	`<`
>	is represented by	`>`

Such entity definitions should be used in all places except within attribute values for tags (tag names and attribute names cannot contain these particular characters). Entity definitions can also be used for special characters, e.g. "é" for a small e with an accute accent. The full list is given in Appendix II. Additional entities may be defined within documents using the SGML entity declaration tag !ENTITY, e.g.

```
<!ENTITY sgml "Standardised General Markup Language">
```

The browser will then insert the full form whenever it comes across "&sgml;".

Repeated white space characters such as space, tab, carriage return, line feed and form feed are ignored except within preformatted text, i.e. it doesn't matter which white space characters you use or how many of them you put between words, or before or after markup elements, the effect is the same as a single space character.

It is strongly recommended that HTML+ documents start with the following external identifier, indicating that the document conforms to the HTML+ DTD. This will ensure that other SGML parsers can process HTML+ documents, without needing to include the DTD with each document.

```
<!DOCTYPE htmlplus PUBLIC "-//Internet/RFC xxxx//EN">
```

HTML+ departs slightly from pure presentation independence by allowing authors to specify rendering hints, e.g. to use a bold font for a given type of emphasis. This step was taken to give authors greater control over the final appearence, and is based upon practical experience with the earlier HTML format. In addition, attribute values are used to distinguish different subcategories of markup, rather than adding extra tags. New logical categories of emphasis etc. can be added in future without needing to change existing browsers. These decisions have made it practical to restrict HTML+ to a very small set of tags.

Backwards Compatibility with HTML

The format is designed to be largely compatible with the earlier format HTML, and it is recommended that HTML+ browsers continue support for the few tags which have been dropped. This will avoid problems for the large numbers of HTML documents without the DOCTYPE declaration. Suggestions on how to map HTML elements to HTML+ are given in Appendix III.

Normal Text

This is generally shown with a serif font and wraps on the right window margin. It can include:

- Entity references, e.g. "`>`" and "`é`"
- Significant Line breaks (the BR tag)
- Hypertext links - the A tag
- Inlined graphics or icons - the IMG tag
- Various styles of logical emphasis - the EM tag
- Embedded data in an external format, e.g. TeX equations - the EMBED tag
- Input fields for forms - the INPUT tag

Line breaks and

Line breaks have a semantic significance in some contexts, e.g. the lines of a poem or a postal address. This tag causes the renderer to start a new line at the current left margin setting. There is no corresponding end tag. The BR tag is *empty*, that is to say, it doesn't act as a container around other text or markup.

Hypertext Links

When the user clicks on a hypertext link in the document, the current document is replaced by the one referenced by the link. Links can be made to a wide range of document types, based on the URL[2] and URN[3] notations. Some document types permit links to be made to specific sections within a document[4]. The syntax for links within the same document or to documents in the same directory is particularly simple:

```
Links are defined with the <a href="#z1">A tag</a>. HTML+ supports a
number of <a href="links.html">different link types</a>.
```

In a browser this might look like:

Links are defined with the *A tag*. HTML+ supports a number of *different link types*.

The first link is to an anchor named "z1" in the current document. The second is to a file named "links.html" in the same directory as the current document. The caption for the link is the text between the start and end tags. The value for the HREF attribute defines the destination point, and can be abbreviated in certain cases. If practical, word the caption in such a way that continues to make sense when the document is printed out. The link should be shown in a clearly recognisable way, e.g. as a raised button, or with underlined text in a particular color. For displays without pointing devices, it is suggested that a reference number is given in square brackets, which can then be typed by the user.

A more general discussion of hypertext links and their treatment in HTML+ is presented in a later section.

Inlined Graphics or Icons

These are treated like characters and inserted as part of the text, e.g.

```
This line has a egyptian hieroglyph at the end of the
line. <img src="ankh.tiff">
```

The URL notation is used to name the source of the graphics data. The *align* attribute can be used to control the vertical position of the image relative to the current text line in which the IMG element is placed. Use a value of "top", "middle" or "bottom" to align the top, middle or bottom of the image with the current text line.

---

[2]The notation for universal resource locators is defined in [Berners-Lee 93b].

[3]The notation for universal resource numbers and the protocol for resolving them to the nearest available copy is currently under study by the IETF URN working group.

[4]At the time this document was written, such links were restricted to named anchors within HTML and HTML+ documents

The *seethru* attribute allows authors to include a chromakey, i.e. a colour that designates portions of the image to be left unpainted so that the background shows through. The format for this attribute's value is dependent on the type of graphics data, and has yet to be defined.

Note that you can create simple iconic buttons, e.g.

```
<a href="bigpic.gif"><img src="littlepic.tiff"></a>
```

If the user clicks anywhere on the image, this will cause the browser to retrieve its bigger version. This approach allows users to preview images which may take significant time to download. Note that there is little additional penalty for displaying the same image at multiple points in the document. The *ismap* attribute is provided for backwards compatibility with HTML. When present the browser will send all mouse clicks and drags on the image, to the server.This mechanism is explained in more detail for the FIG tag.

Sophistocated HTML+ editors should allow authors to modify images using an external editor. Larger images should be specified with the FIG tag.which provides support for flowing text around figures, along with captions, overlays and active areas.

Various Styles of Emphasis[5]

This allows you to emphasise a portion of the text. The simplest approach is:

```
<em>default emphasis, usually shown in an italic font</em>
```

The logical role of emphasis denotes the semantic significance, e.g. a citation, or text to be input by a user for a computer program.The physical style of emphasis controls its appearence. Note that EM elements can include inlined graphics.

Logical Role of Emphasis

It is strongly recommended that the logical role of the emphasis is given with the *role* attribute, e.g.

```
<em role="cite">a citation</em>
```

Providing a logical role allows browsers to apply differing rendering styles according to the role, but more importantly, it allows indexes to be constructed automatically, e.g. the list of bibliographic references in a technical report. These can be used for searching through collections of documents according to semantic keys giving better focussed searches compared with full text indexes .

The list of recommended roles are as follows: *(this can be given in upper or lower case)*

For references to other works:

cite	a reference to a related work
pub	a publication containing a referenced work
author	an author of a referenced work
editor	an editor of a referenced work
title	the title of a referenced work
credits	e.g. the rights owner of a photograph
copyright	the holder of the copyright
isbn	for ISBN numbers
acronym	for acronyms like "NATO" and "US"
abbrev	for abbreviations

For annotations:

footnote	shown as footnote or pop-up
margin	shown as margin note or pop-up

For computer instruction manuals:

dfn	defining instance of a term

[5]The name EM was chosen in preference to EMPH because it allows existing HTML browsers to show all HTML+ emphasis in italics. It also allows HTML+ browsers to correctly process the common case for emphasis in HTML documents.

kbd	something a user would have to type
cmd	command name, e.g. "chmod"
arg	command arguments, e.g. "-l"
var	named place holder, e.g. "filename"
ins	an instanced of a named printer, directory or file etc.
opt	an option of some kind
code	an example of code (shown with a fixed pitch font)
samp	a sequence of literal characters

On dumb terminals annotations should be shown in round brackets. Margin notes should be right aligned, and may include graphics via the IMG tag. The set of recommended roles will be kept by the HTML+ registration authority.

Physical Styles

The appearence can be modified by adding optional rendering hints from the list:

```
<em b>              bold text
<em i>              italic text
<em u>              underlined text
<em sup>            superscript text
<em sub>            subscript text
<em tt>             type writer font (courier)
<em hv>             sans serif font (helvetica)
<em tr>             serif font (times roman)
```

These hints can be combined, e.g.

```
<em b i> for bold italic text </em>
```

Note that these are only hints and may be ignored by browsers. Indeed, arbitrary combinations will present difficulties for most browsers. If the display is limited to a single font, colour or underlining can be used, but should be clearly differentiated from hypertext links and headers. Dumb terminals can use email conventions, e.g. switching to all capitals, or delimiting with the * or _ characters. Subscript and superscript text should be shown in a smaller point size, vertically offset as appropriate.

Browsers may choose to simplify or ignore hints, but should aim to do so in a consistent manner. At the simplest level, browsers can ignore the attributes and render all emphasis in the same style.

Nested Emphasis

Emphasis can be nested as in:

```
<em b>bold text, and <em i>bold italic text</em></em>
```

Nested emphasis is better suited for grouping logical roles together, for instance, you could use EM to separately tag author, title, and publication, and then wrap these up as a citation. Without this, indexing programs will have difficulty in grouping markup into the correct references.

Horizontal Rule

The <HR> tag may be used to draw a horizontal rule to separate text sections. It can be rendered as a simple line across the middle section of the window/paper or similar decoration.

Embedded data in an external format

The EMBED tag provides a simple form of object level embedding. This is very convenient for mathematical equations and simple drawings. It allows authors to continue to use familiar standards, such as *TeX* and *eqn*. Images and complex drawings are better specified using the FIG or IMG elements. The *type* attribute specifies a registered MIME content type and is used by the browser to identify the appropriate shared library or external filter to use to render the embedded data, e.g. by returning a pixmap. It should be possible to add support for new formats without having to change the browser's code, e.g. through using a common calling mechanism and name binding scheme. Sophistocated browsers can link to external editors for creating or revising embedded data. Arbitrary 8-bit data is allowed, but &, < and > must be replaced by their SGML entity definitions. For example <embed type="application/eqn"> 2 pi int sin (omega t)dt </embed> gives

$$2\pi\int \sin(\omega t)dt$$

Input Fields for Forms

Input fields can be arranged with considerable freedom, as part of normal paragraphs, preformatted text, lists or tables. Examples of how to do this are given later on in the section describing the FORM tag. The INPUT tag has the following attributes:

name Used to name this input field, e.g. `name="phone number"` *(required attribute).*

type Defines the type of data the field accepts (the type name is insensitive to upper/lower case). If missing, the field is assumed to be a free text field.

size Specifies the size/precision of the input field according to its type, see below *(optional).*

value The initial value for the field, or the value when checked for checkboxes and radio buttons *(optional, except for radio buttons).*

checked When present, this attribute indicates that a checkbox or radio button is selected.

disabled When present, this attribute indicates that this field is temporarily disabled. Browsers should show this by greying out or via a similar visual clue. Users are unable to set the focus to disabled fields, or change their values.

error When present, this attribute indicates that the current value for this field is in error in some way, e.g. because it violates some consistency constraints. Browsers should indicate this by a change to the shape and colour (red) of the field's border. This should be accompanied by an error message and a beep.

The following types of field should be supported: *(in either upper or lower case)*

text Single or multi-line text entry fields. Use the *size* attribute to specify the width and height in characters, e.g. size="24" or size="32x4".

url For fields which expect document references as URL or URNs.

int For entering integer numbers, the maximum number of digits may be given with the *size* attribute, e.g. size=3 for a 3 digit number[6].

float For fields restricted to floating point numbers.

date Restricted to a recognised date format.

checkbox Use these for simple boolean attributes, or for attributes which can take multiple values at the same time from some set of alternatives (for fields with the same *name*).

radio Use these for attributes which can take a single value from a set of alternatives (groups input fields with the same *name*).

For the purposes of sending the contents of a form to a server, as part of a query, the input fields are mapped to a list of properties. In most cases the *name* and current *value* are used to define a property/value pair for each field. Radio buttons and check boxes are left out of the list if they are unselected. This ensures that only the selected radio button yields a property/value pair. By missing out the *value* attribute for check boxes, these fields will map to a simple (value-less) property. The representation of property lists is defined as part of the HTTP protocol.

Browsers can choose to notify the server whenever a field is changed (i.e. when a field looses the focus and its contents have changed) or wait until the form is completed. This choice will depend on network latency.

Drop down or "combo" style selection lists may be added in a future revision to this standard.

---

[6]Perhaps the syntax should permit integer ranges, e.g. size="1 to 6", in which case a more appropriate name for the attribute than *size* would be desirable.

Headers and Titles

The title tag is generally used to define the window banner when viewing a particular document, e.g.

```
<title>Reference Guide to HTML+</title>
```

This element should appear at the start of the document. There are six levels of headers, H1 to H6, with H1 the most important, and H6 the least. A common convention is to begin the body of the document with a level one header. e.g.

```
<h1>Introduction to HTML+</h1>
```

Header names should be appropriate to the following section of the document, while the document title should cover the document as a whole. There are no restrictions on the sequence of headers, e.g. you could use a level three header following a level one header. Browsers should render headers with a line break before and after the header text. A common convention for headers is to use a sans serif font, e.g. Helvetica, with a smaller point sizes for less significant headers, and a serif font, e.g. Times Roman, for normal text.

Headers can include an identifier, unique to the current document, for use as destinations of hypertext links, e.g.

```
<h1 id="intro">Introduction to HTML+</h1>
```

This allows authors to make links to particular sections of documents. It is a good idea to use something obvious when creating an identifier, to help jog your memory at a later date. WYSIWYG editors may automatically generate the identifiers. In this case, they should also provide a point and click mechanism for defining links, so that authors don't need to deal explicitly with the identifiers.

The attribute "margin" when present acts as a hint to the browser to insert the header into the margin and causes the following text to be vertically aligned with the start of the margin header. By convention, margin headers are left justified, e.g.

```
<h4 margin> Deleting the Curve </h4>

The Delete command allows you to delete any selected symbol or text
block.
```

Note that headers don't act as containers for the subsequent text. You can group the header and text with the GROUP tag, see later for details.

Indexing

A good index plays an important role in helping you find your way to the material you need. It allows you to type in one or more keywords to see a list of matching topics. Alternatively you can browse through the index and take advantage of serendipity. This also allows you to gain a feeling for the limits of what is covered. The two approaches can be combined, when the characters typed act dynamically to control the viewing position within the index.

Typically each keyword entry in the index is associated with one or more topics. This notion of guiding the user is absent from full text indexes like WAIS, where users are given very little help in choosing the keywords to search on. Generating a conventional index for a document is a skilled task, and HTML+ allows authors to include directives for creating an index. These directives can be included with document titles, headers and emphasis etc. using the *index* attribute. This allows each such element to be included in one or more entries in the index, under primary or secondary keys, e.g.

```
<h3 id="z23" index="Radiation damage/shielding from as difficult">Radiation shielding</h3>
```

This resulting index looks like:[7]

Radiation damage
  classical target theory
  dominance of
  in molecular mills
  shielding from as difficult
  simple lifetime model
  track-structure lifetime model
Radicals
  and so on.

Where each entry is a hypertext link to the associated anchor. The *index* attribute can specify multiple entries, each separated with the ";" character. The optional secondary key (*shielding from as difficult*) is introduced by the "/" character. Secondary keys are useful when the primary key occurs more than once. To allow for future extension, primary keys should not start with the "#" character. This prefix is being reserved to designate indirect index entries. Use "\/", "\;", "\#" and "\\" to escape "/", ";", "#" and "\" respectively.

Paragraphs and Preformatted Text

HTML+ includes support for paragraphs and preformattted or verbatim text.

Defining Paragraphs with <P>

The P tag is used to define paragraphs. Unlike many other tags, the end tag is optional. Note, that unlike HTML,the tag acts as a container for the text of the paragraph. This allows paragraphs to act as hypertext anchors. The end of the paragraph is implied by finding markup elements which are not permitted as part of a paragraph.

The following attributes may be used:

id	An identifier, unique to this document, which can be used as a destination in a hypertext link.
role	The role of the paragraph, see the following list for supported types.
align	A rendering hint to the browser to justify lines. The supported values should be: `align="left"`, `align="center"`, `align="right"` and `align="flush"`. This is useful for single line paragraphs or when the lines are made explicit with the tag.
indent	When present, this hint suggests that the left and right margins are indented by an amount dependent on the browser, e.g. about 4 character widths.

The *role* attribute is used to indicate the logical role of the paragraph, e.g. a stanza in a poem or a cautionary note in a computer manual. Browsers may apply particular rendering styles to certain roles. The role name is case insensitive.

The following roles are recommended: *(in upper or lower case)*

quote	A paragraph quoted directly from some other work. Browsers could indent the paragraph and maybe use a different font.
byline	Information about the author of the document, e.g contact details. This could be displayed in a different font, and perhaps right aligned.
note	Advisory note in an instruction manual. The browser could display a hand icon in the margin.
caution	Cautionary note. The browser could display an warning road sign in the margin.
error	A note describing error conditions. The browser could indicate the importance of the note by displaying a stop sign in the margin.

[7]Taken from K. Eric Drexlers's "Nanosystems, Molecular Machinery, Manufacturing and Computation".

An example of a paragraph element:

```
<p role="note"> If you accidentally delete a symbol other than the red
circle, immediately press ALT+BKSP to choose the undo command, and
then select the red circle and delete it again.
```

Paragraphs can be rendered by indenting the first line, or by leaving a vertical gap, for example, half the current line spacing. When using the latter style, browsers should take care to avoid inserting this vertical gap when the paragraph element immediately follows a header. This rule ensures that authors can tag paragraphs directly following a header without causing unwanted extra space to appear before the start of the text.

Preformatted Text with <PRE>

This is generally shown in a fixed pitch font and preserves the original spaces and line breaks. The horizontal tab character is deprecated, but should be interpreted as a move to the next tab setting, at every eighth column. Preformatted text is useful for including plain ASCII text, e.g. program listings and email messages. A number of tags can be included within preformatted text, e.g. hypertext links using the A tag, emphasis, inline images and input fields. The following optional attributes can be used:

id An identifier, unique to this document, which can be used as a destination in a hypertext link. Note that the paragraph tag acts as a container for the paragraph.

role The role of the element.

tr Use a proportional serif font, e.g. Times Roman.

hv Use a proportional sans serif font, e.g. Helvetica.

width This gives the maximum number of characters which will occur on a line. The default value is assumed to be 80. Browsers recognising this attribute should optimally handle widths of 40, 80 and 132, with other widths being rounded up.

Preformatted text started off in HTML with a simple mechanism for showing computer output, for which the spaces and line breaks were significant in determining the layout. The desire to render Unix manual pages as hypertext forced a rethink. The new version supported character emphasis and hypertext buttons for cross references. HTML+ adds the capability to use variable pitch fonts, and to set up tab stops, e.g.

```
<tab at=40 align=right>
```

The *at* attribute specifies the position of the tab stop, measured from the left margin in terms of the width of the character M for the current font. The *align* attribute is one of "left", "center", "right" or "decimal", defaulting to left alignment. For greater control of layout, authors should exploit the FIG or EMBED tags to use external formats, for example encapsulated Postscript. Unfortunately these formats don't as yet support hypertext links.

Ordered, Unordered and Definition Lists

There are three kinds of lists: ordered or numbered lists, unordered lists and definition lists. Ordered and unordered lists can be nested arbitrarily, and browsers should progressively inset the left margin for each level of nesting.

Ordered Lists with <OL>

The list items are automatically numbered, e.g.

```
<OL>
  <LI>Wake up
  <LI>Get dressed
  <LI>Have breakfast
  <LI>Drive to work
</OL>
```

Is displayed as:

1) Wake up
2) Get dressed
3) Have breakfast
4) Drive to work

The *compact* attribute when present, e.g. <ol compact>, has the effect of reducing interitem spacing. Authors can also make both the OL tag and the LI tag potential destinations for hypertext links with the *id* attribute. List item text can include normal text and paragraph elements, but not headers.

Unordered Lists with <UL>

These are bulleted lists, e.g.

```
<UL>
  <LI>Wake up
  <LI>Get dressed
  <LI>Have breakfast
  <LI>Drive to work
</UL>
```

Is displayed as a bulleted list:

- ❑ Wakeup
- ❑ Get Dressed
- ❑ Have breakfast
- ❑ Drive to work

The *compact* attribute when present, e.g. <ul compact>, has the effect of suppressing bullets and reducing interitem spacing. Multicolumn lists can be requested with the *narrow* attribute, e.g. <ul narrow>. This causes the browser to try to lay out the list as a number of columns, depending on the window width. This attribute should only be used when all the items are less than 20 characters long. Authors can also make both the UL tag and the LI tag potential destinations for hypertext links with the *id* attribute. List item text can include normal text and paragraph elements, but not headers. For nested unordered lists, browsers may use different bullet symbols for different levels, in addition to progressively insetting the left margin. The *src* attribute on the LI tag can be used to specify an icon for use in place of the standard bullet symbols.

Definition Lists with <DL>

These consists of pairs of terms <DT> and definitions <DD>. The following example is part of a french dictionary:

```
<DL>
  <DT>endetter
  <DD>Engager dans des dettes

  <DT>endeuiller
  <DD>Plonger dans le deuil, remplir de tristesse

  <DT>endiablé, ée
  <DD>D'une vivacité extrême
</DL>
```

Is commonly displayed as:

endetter	Engager dans des dettes
endeuiller	Plonger dans le deuil, remplir de tristesse
endiablé, ée	D'une vivacité extrême

With the *compact* attribute, e.g. <dl compact>, this could be altered to:

endetter Engager dans des dettes

endeuiller Plonger dans le deuil, remplir de tristesse

endiablé, ée D'une vivacité extrême

In this style, the term and definition appear in the same paragraph, with the term text emphasised in a bold font. The definition text follows on, and wraps to a left margin a little further inset than the term text. This style is common place in dictionaries.

Term text following the <DT> is restricted to normal text. The definition text after the <DD> tag can additionally include paragraph elements and ordered/unordered lists. Headers are not allowed in either case. Authors can make the DL, DT and DD tags potential destinations for hypertext links with the *id* attribute.

Authors are reminded to check that DT and DD are paired up. Common misunderstandings lead to people repeating DD tags to separate paragraphs (use <P> instead), or leaving out the DT tag altogether to indent text (use <p indent> or <group indent>). The ability of browsers to cope with bad markup seems to encourage such problems, which will hopefull fade away as wysiwyg editors become commonplace

Figures

Figures provide great flexibility, and can be used to show images, graphics or other information specified in an external format:

- linked or embedded definitions
- control of picture alignment and text flow
- Figure description for when the image can't be shown
- caption placement
- scaled or pixel-based coordinates
- hypertext links with active areas
- text and image overlays

The following simple example will set the scene for the description of the various features:

```
<fig align="right" src="map.gif"> How to get to my house </fig>
```

Here, the image is defined by a link to an external document. The caption "How to get to my house" will appear at the bottom of the image. The *align* attribute directs the browser to display the figure at the right of the widow, and to flow subsequent text around the left of the image.

Using embedded graphics data

Instead of the *src* attribute, you can include an EMBED element immediately following the <fig> tag. This is useful for simple graphs etc. defined in an external format.

Figure Description

The FIGD tag allows you to give a textual description which can be shown when the figure itself can't be shown, e.g. for browsers working on dumb terminals, e.g.

```
<FIGD> This is an aerial photograph of central London, showing
Buckingham Palace and the Houses of Parliament. On the left you can see
Hyde Park and in front the Albert Hall and the Natural History
Museum.</FIGD>
```

Alignment and Text Flow

The *align* attribute controls the horizontal position of the figure: "left", "right", or "center". The default is "left". Browsers may flow text when there is sufficient room, unless the figure is center aligned or the *noflow* attribute is present.

Caption Placement

The *cap* attribute allows you to ask the browser to position the caption text to the "left", "right", "top" or "bottom". The default is to place the caption at the bottom of the figure. Text flow will occur around the figure and caption, leaving a suitable gully. The browser will ignore this attribute if there is insufficient room for the requested placement.

Pixel-base or Scaled Coordinates

The upper left of the figure is designated as x,y = (0, 0), with x increasing across the page, and y down the page. If points are given in real numbers, the lower right is taken as being (1.0, 1.0), otherwise with integer values, the coordinates are assumed to be in pixels[8]. Note that using scaled coordinates is much safer, especially for graphics! The extent of the image in pixels may change, e.g. as a result of format negotiation with the server, and by retrieving images with lower resolution when network performance is poor.

Active areas

The *ismap* attribute causes the browser to send mouse clicks on the figure, back to the server using the selected coordinate scheme. The mouse button-up event is sent with the URL formed by adding "?x,y" as a suffix to the URL for the current document. You can also designate rectangular regions of interest in the picture by holding the mouse button down while dragging the mouse. The browser should show a rubber band outline for the rectangle defined by the current location of the mouse pointer and the point at which the mouse button was pressed. The region is named by taking the current URL and adding the suffix: "?x1,y1;x2,y1;x2,y2"[9], where (x1, y1) and (x2, y2) define the points at which the mouse button went down and came up, respectively. The *ismap* mechanism is relatively slow, but makes sense when the active regions change their boundaries over time, e.g.

```
<fig ismap src="weather.gif">Click on your area for todays weather</fig>
```

You can also designate arbitrary areas of the figure as hypertext links. Mouse clicks are handled locally, and the browser can provide visual clues that the pointer is over an active area, for example, by changing the pointer from an arrow to a hand symbol, or highlighting the area in some way.

Active areas are defined with the FIGA tag. This has two attributes:

href	A URL specifying the link to traverse when clicked (required)
area	Defines a polygonal[10] area as a list of points: "x1, y1; x2, y2; ..." (optional)

The *area* attribute lists a sequence of points defining a polygon. Closure is ensured by joining the last point in the list to the first (i.e. a triangular area is defined with a list of 3 points).When the *area* attribute is missing, the whole of the picture is assumed. Polygons may be non-convex or even intersect themselves, thereby complicating the definition of what is enclosed by the polygon. Holes should be excluded. Note that active areas defined with FIGA take precedence over the *ismap* mechanism.

Overlays

The FIGT tag allows you to position text and image overlays on top of the figure, e.g.

```
<fig src="map.giff">
  <figt at="0.2, 0.3" framed>A text overlay</figt>
  The figure caption
</fig>
```

---

[8]This mechanism was designed to be backwards compatible with the *ismap* feature as used with IMG in HTML, and as a consequence forces the choice of y increasing down rather than up the page. A simple test to distinguish the two schemes is to check if the "." character occurs anywhere in the list of points.

[9]This definition is intended to allow future extension to arbitrary polygons, and hence is chosen to be directly compatible with the *area* attribute of the FIGA tag.

[10]The code for hit testing polygons is tricky, but quite fast. A public domain version of the code would be helpful.

The overlay can contain a wide variety of elements including text, images (IMG), lists and tables. Figures shouldn't be nested. Any hypertext links in the overlay text will take precedence over the *href* attribute in FIGT. The following attributes are permitted:

idref Names the FIG element to which this overlay applies. This is only relevant when overlays are held separately as a form of annotation. This attribute then allows the annotation to be correctly merged back into the document.

at The upper left of the overlay, relative to the figure.

width As a fraction of the figure, e.g. width="0.3". This allows you to limit the lengths of wrapped text lines. The vertical extent is then determined automatically.

framed Directs the browser to draw a frame around the overlay and to colour in the background in some way.

href Allows you to make the overlay into a hypertext button.

Tables

Tables are defined with the TBL tag. Cells are designated as being headers or data. You can join adjacent cells, e.g. to define a header spanning two columns.

An Example of a Table

	average		other
	height	weight	category
males	1.9	0.003	yyy
females	1.7	0.002	xxx

This is defined by the markup:

```
<tbl border>
   <tt top> An Example of a Table
   <th rowspan=2> <th colspan="2"> average <th> other <tr>
   <th> height <th> weight <th> category <tr>
   <th align=left> males <td> 1.9 <td> .003 <td> yyy <tr>
   <th align=left> females <td> 1.7 <td> .002 <td> xxx
</tbl>
```

The *border* attribute for TBL directs the browser to draw borders. The *compact* attribute is used when you want the table to appear in a smaller size.

The optional <tt> tag defines a title. By default (i.e. when *top* is missing) this should be positioned below the table. The <th> and <td> tags define header or data cells respectively. The <tr> tag acts as a separator between rows. In the example, you can see that the first header in each of the first two rows is void.

TH, and TD all have the same permitted attributes:

colspan Columns spanned by this cell, see example

rowspan[11] Rows spanned by this cell, see example

align=left Left justify the cell's content

align=center Center justify the cell's content

align=right Right justify the cell's content

[11]This is tricky to handle. The parser should carry a spanned cell over to the next row, the definition of which should miss out the spanned cell, i.e. the next row will have one fewer explicit cell definitions.

By default, headers are centered, while other cells are left justified. If practical, browsers should be smarter than this, e.g. if all the cells in a column are shorter than the column header, then indent the cells to make them appear under the middle of the header.

Browsers need to carry out a pre-parse (e.g. when sizing the vertical scroll bar) in order to determine the number of columns and their widths. The following guidelines may be useful:

- There is no need to declare empty cells at the end of a row, so the number of columns for the table is given by the row with the most columns.
- Restricting text to a fixed pitch font may simplify matters.
- If a column only contains numbers or empty cells then align on units and set width to the maxium precision needed (before and after decimal point, allowing for an exponent). This rule also applies when currency symbols are used.
- Otherwise set column width to the minimum of a threshold width and the maximum text length for all cells in the column. Text is left aligned and wrapped if it exceeds the chosen column width.

The threshold column width can be set according to the number of columns and the width of the display window. It is also necessary to take the column headers into account in this process. Header text wraps to the next line if the column is too narrow. Browsers will by default center the header in the column.

A complication occurs when a header or data cell spans more than one column, as specified by the *s* attribute. This can be used to give complex headers which share a header between columns followed by individual headers on the next line.

Vertical gaps can be introduced with the <tb> element - this inserts 1/2 line space into the next row. Header and Data rows can be intermixed. Authors can use alternate header and data rows when the rows alternate between text and numbers. The vertical alignment of numbers only applies to data fields.

Tables which don't fit into this model should be defined as figures using an external format, e.g. Postscript, TeX or Computer Graphics Metafile.

Forms

A document can include one or more forms. Each form is defined by a FORM element, which contains a number of input fields laid out with normal and preformatted text, lists and tables. The browser should manage the input focus, e.g. with the tab key and mouse clicks. The Return key can be used to mean that the user has filled in the form and wants the appropriate action to be taken. Browsers may also display "Accept" and "Cancel" buttons as part of the document (or perhaps on another part of the browser). Note that forms shouldn't be nested.

The action to be taken is specified by the *action* attribute of the FORM tag. If missing the URL for the current document is assumed. This attribute uses a URL to specify a server to query, or an email address to send the form to. When sending the form to a server as a query, the form's contents are encoded as a property list (see definition of the INPUT tag). The precise encoding is dependent on the HTTP protocol and defined in [Berners-Lee 93c][12]. When the form is to be mailed, it is first converted into plain text, closely resembling the appearence on the screen. You can include multiple RFC 822 mail headers with the MH tag. The *hidden* attribute may be used to hide the headers when browsing the document. The following is an example of a simple questionaire:

```
<form action="mailto:www_admin@info.cern.ch">
<mh hidden>
  Subject: WWW questionaire
</mh>
Please help us to improve the World Wide Web by filling in the
following questionaire:
<p>
Your organisation? <input name="org" size="48">
```

[12]This and the *ismap* feature rely on the forthcoming definition of HTTP as an official Internet standard.

```
<p> commercial? <input name="commerce" type="checkbox">
How many users? <input name="users" type="int">
<p> Which browsers do you use?
<ol compact>
<li> X Mosaic <input name="browsers" type="checkbox" value="xmosaic">
<li> Cello <input name="browsers" type="checkbox" value="cello">
<li> Viola <input name="browsers" type="checkbox" value="viola">
<li> Others? <input name="other browsers" size="48x4">
</ol>
A contact point for your site: <input name="contact" size="48">
<p>Many thanks on behalf of the WWW central support team.
</form>
```

Floating Panels

The PANEL tag can be used to define panels or boxes which are free to float with respect to the standard flow of text. These are often used in magazine articles for asides on background material. The panel is typically shown with a distinctive background colour and border. The lay out software positions the panel to coincide with the page boundaries in printed media. For on-line use, panels can be rendered as pop-up windows. The body of the panel can be defined by a link to a separate document or included in the current document.

Another application of the panel tag is for annotations by one or more reviewers. The annotations can be held separately e.g. with an annotation server and subsequently retrieved and merged with the document. The annotation server returns a list of annotations in response to being queried with a URL or URN. This list can be expressed as a multi-part MIME message with the author and date passed as RFC 822 headers for each annotation.

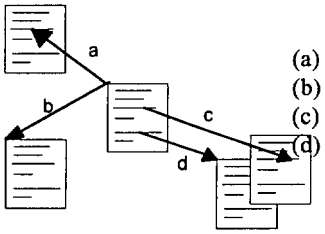
The following optional attributes are permitted with the <panel> tag:

id	An identifier, unique to this document, which can be used as a destination in a hypertext link.
at	An identifier elsewhere in this document. The panel mustn't be placed before this point. (Defaults to the current position if the *at* attribute is missing).
role	This may be used to clarify the role of the panel, e.g. as an "annotation" or an "aside".
href	This attribute allows authors to fill the panel from a separate document, as specified by a URL. Note that the matching end tag: </panel> is always needed.

The text contained by the panel element can include any of the markup elements and looks like a separate document (panels themselves can't be nested). If the *href* attribute is used the text delimited by <panel> ... </panel> may be used as the caption for a pop-up. The *at* attribute allows you to include the panel definition at a convenient point in the HTML+ document, rather than interrupting the main flow of the document.

More on Links

Before describing the details of how links are represented in HTML+ it is worth looking more generally at the nature of hypertext links. First a terminological point: a *node* is the atomic unit for information retrieval, while *documents* may consist of one or more nodes, perhaps arranged as a hierarchy. A node may even be shared between several documents. Hypertext links start and end on nodes or anchors, where anchors specify portions of nodes, e.g. a paragraph or list.



The diagram illustrates the basic possibilities:

(a)	Link from a node to an anchor
(b)	Link from a node to a node
(c)	Link from an anchor to another anchor
(d)	Link from an anchor to a node

Links in HTML+ are represented with the LINK and A tags. The LINK tag is used for cases (a) and (b), while the A tag is used for cases (c) and (d). These links are held in the source node only, so there is a risk that the destination may have disappeared. Organisations can manage this risk by long term support for a few well published nodes (servers can use redirection to hide internal name changes for these nodes). Links to other subsidiary nodes are at higher risk. This structured approach allows people to become familiar with the major routes through the web, without needing to worry about the minor routes.

In most cases URLs and URNs explicitly specify a node/anchor. The nodes may be explicit files or generated as the result of some process invoked by the server, e.g. a hypertext listing of a directory or a list of matches for a given search string. The search string can be explicitly encoded as part of a link, or dynamically defined by the user (see the ISINDEX tag, as described later on).

Links may be held separately from the source and destination nodes[13]. This is particularly appropriate for annotations and discussion groups. For example, consider making an annotation on a document held by a server located far away in another organisation. You could take a local copy and directly annotate it, but this is only appropriate for private use. The remote server might even support a protocol to add your annotations in place. More likely though, you will have to use an annotation server. This mechanism can be used to obtain a copy of the document with the annotations inserted as hypertext links and shown as pop-ups or separate documents.

Context Dependent Links

For discussion groups, responses are made asynchronously, and include one or more references to other articles. In this situation, context dependent links are appropriate. The resolution to an explicit node can be carried out by either the client or server. The former approach is often appropriate, but requires special support in the browser, e.g. for network news and nntp.

Context dependent links are also useful for links to the table of contents for documents consisting of multiple nodes, when some of the nodes also appear in other documents. The appropriate table of contents for a given node will depend on which document is currently being viewed. In this case, the context will depend on how the current node was reached.[14] This is quite simple to track if the links from the table of contents are differentiated from cross reference links.

Hypertext paths are recommended routes through a set of nodes, and generally shown by next and previous buttons on a toolbar. Paths can be defined using explicit links in a node, or held separately in another node. The latter case once again, depends on the context. Paths and tables of contents all fall under the general category of navigating around a hierarchy of nodes forming a document too large or unwieldy to be held in a single node.

Types of Links

There are several motivations for differentiating between types of links:

how it is viewed	The potential to show different cues depending on the type and size of the node to be retrieved. If this information is explicitly stated as part of the link, there is a chance that it will become out of step with the linked node.
what happens	Whether the linked document replaces the current one, or appears in a new window, or as a pop-up overlay on top of the current one.
printed appearance	Whether links are treated as references, footnotes or as separate sections
effect on context	After traversing the link, will there be implicit values for the table of contents, and hypertext path etc?

[13]These correspond to HyTime's *ilink* architectural form.

[14]This is more general than deriving the role of the link from that of the node alone.

Link Attributes

The A tag has the following attributes:

id	An identifier unique to this document which can act as a hypertext anchor
name	The same as *id* and included for backwards compatibility with HTML. New documents should use the *id* attribute for consistency with the other tags.
href	The URL or URN identifying the destination of the link.
role	A string giving the role of the link, e.g. role="partof" or "annotation"
effect	A string defining how the linked node is shown: "replace", "new", "overlay", with the default effect of replacing the current document.
print	How should the link be printed: "reference", "footnote" and "section", defaulting to "reference" (i.e. a footnote stating the link's URL).
title	The title to show when otherwise undefined for the node.
type	The MIME content type for the linked node for use with presentation cues.
size	The size in bytes for the linked node. This allows the browser to show a gauge indicating progress in retrieving long documents or images etc.

The LINK tag has only the *href* and *role* attributes.

The *role* attribute is appropriate when context dependent properties such as *table of contents* (toc) are implied for the linked node, e.g. if the current node is a toc (as defined by the *html* or *group* tags) and the link has the role "partof", then the current node should act as the *toc* for the linked node. This property propagates down "partof" links, but not normal links. The *next* and *prev* properties are given by the sequence of "partof" links in the parent node. The *parent* property is only defined if the current node was reached via a "partof" link.

The LINK tag is used to express these properties in an explicit form, e.g.

```
<LINK href="toc.html" role="toc">
```

The recommended property names are: (in upper or lower case)

toc	Table of contents for current node.
next	The next node in a hypertext path.
prev	The previous node in a hypertext path.
parent	The next level up in the hierarchy.
index	A searchable index appropriate to this node.
style	The style sheet appropriate to this node.

Style sheets provide a way for authors to express their detailed preferences for fonts, and layout, whether for the screen or when the node is printed out. A possible format is given in [Raisch 93].

The *effect* attribute is a hint and may be disregarded by browsers. It allows you to click on an image and to see a linked movie as an overlay at the same position. The browser tries to position the overlay at the same origin as the link. In some cases, the linked node is a description of the current node. By including `effect="new"`, the linked node will appear in a new window so that users can see both nodes at the same time. This hint should be used sparingly!

The *print* attribute makes it practical to print nodes along with relevant linked nodes. By default each link appears as a footnote stating the link's URL. Short nodes can be included in their entirety as footnotes, and longer ones as sections in their own right. This approach could be extended in future, to reorder the sequence of nodes from that defined by the position of the links in the source node, and to control the level that nodes appear as, e.g. chapter, section or subsection.

The *title* attribute is useful for nodes without titles of their own, e.g. Gopher menus. The *type* attribute can be used to show cues for the node type, e.g. iconic decorations[15]. The *size* attribute allows browsers to show a gauge on how much of a document has been retrieved at a any time. These attributes are liable to get out of step with the target node, and should be treated as hints only.

Groups

The GROUP tag allows you to define arbitrary groups, e.g. books, chapters, and sections. The *role* attribute is used to name the logical role of the group. You can use most markup elements inside a group element, including group itself. The *inset* attribute is a rendering hint to inset the left margin. Using the A tag with role="partof" allows you to designate a node as being included within the group, allowing hierarchies of groups which cross multiple nodes. See previous discussion of how properties are propagated.

Groups offer opportunities for presenting and searching documents at different levels of abstraction. For example, you might first describe a book by its title, author, publisher and ISBN number. The next level down could add a cover illustration together with a summary of the book's contents, some comments by reviewers and a short biography of the author. This would allow a list of books to be presented in an iconic form using a miniature version of the "cover page". Publishers could include copyright and other details in a standard place.

Change Bars

Authors can indicate a part of a document has been changed using the CHANGED tag. This may appear anywhere that normal text is allowed (as designated by the entity reference `%text;` in the DTD). This tag signals the beginning or end of changes, which should be rendered by a vertical bar in the left margin. The tag can have one (but not both) of the following attributes:

id An identifier unique to the current document, which can also be used as a a destination for hypertext links. This signals the beginning of changes, e.g. `<changed id=z34>`.

idref This must be an identifier matching the preceding changed element. It signals the end of changes. Note that you mustn't have both *id* and *idref* together, e.g. `<changed idref=<z34>`.

Notes

The NOTE tag allows authors to name an arbitary portion of a document. It can be used much more freely than the A tag which is restricted in the markup it can contain. The NOTE tag is intended for delimiting the portion of a document associated with an annotation. The annotation itself is linked via the *href* attribute or expressed as a panel element, whose *at* attribute names this note. The latter allows several annotations to apply to the same place.

The permitted attributes are:

id An identifier unique to the current document, which can also be used as a a destination for hypertext links. This signals the beginning of the note, e.g. `<note id=z34>`.

href May be used to directly specify an associated annotation with a URL.

idref This must be an identifier matching the preceding note element. It signals the end of changes. Note that you mustn't have both *id* and *idref* together, e.g. `<note idref=z34>`.

---

[15]The appropriate cue might also depend on the role of the link, e.g. for annotations browsers could show an icon of a drawing pin (as in attaching a note to a pin board). The colour of the pin could then vary according to the media type of the annotation.

Miscellaneous Tags

The remaining tags must appear at the start of the node like TITLE and LINK. They describe properties which apply to the node as a whole.

The HTML tag.

This is intended to provide short informal classifications for use in cataloging documents held by HTTP servers. The *role* attribute identifies the purpose of the node, for example `<html role="home page">`. Another common role is "toc" for table of contents. See previous discussion of link attributes.

The ISINDEX tag

This specifies that the URL designated with the *href* attribute is searchable (defaults to this document's URL). Browsers should allow users to enter a search string of one or more keywords. When the Return key is pressed the search string is appended to the designated URL, after a "?" character and sent to the server specified by the URL. Certain characters should be escaped as specified by the standard URL syntax, for example, the space character is mapped to "+". The newer HTTP protocol offers an alternative means for specifying that documents are searchable. See [Berners-Lee 93c] for details.

The NEXTID tag

This is used by browsers that automatically generate identifiers for anchor points. It specifies the next identifier to use, to avoid confusion with old (deleted) values, e.g. `<nextid n="id56">`. The identifier should take the form of zero or more letters followed by one or more digits. The numeric suffix should be incremented to generate successive identifiers.

The BASE tag

The *href* attribute gives the full URL of the document, and is added by the browser when the user makes a local copy. Keeping the original URL in a local copy is essential when subsequently viewing the copy as it allows relative URLs in the document to be resolved to their original references.

Note that one motivation for using relative URLs is to allow a group of documents to be copied without the need to alter any links between them. In this case, the BASE tag is inappropriate, since it would cause links to be intepreted as being to the original documents rather than their copies.

The HEAD and BODY tags

The HEAD tag can be used to delimit properties which apply to the document as a whole, and if used, must be present at the start of the document, followed by the BODY tag which then delimits the rest of the document.

Acknowledgements

I would like to thank the many people on the *www-talk* mailing list who have contributed to the design of HTML+ and to the management of HP Labs for their support during this work.

David Raggett, Hewlett Packard Laboratories, July 1993.

Email: dsr@hplb.hpl.hp.com, *Phone*: +44 272 228046

OIPE
OCT 24 2003
PATENT & TRADEMARK OFFICE

Appendix I - The HTML+ DTD

```
<!SGML  "ISO 8879:1986"
--
      Document Type Definition for the HyperText Markup Language
      Plus for use with the World Wide Web application (HTML+
      DTD).
      NOTE: This is a definition of HTML+ with respect to
      SGML, and assumes an understanding of SGML terms.
--
CHARSET
      BASESET     "ISO 646:1983//CHARSET
                  International Reference Version (IRV)//ESC 2/5 4/0"
      DESCSET     0     9     UNUSED
                  9     2     9
                  11    2     UNUSED
                  13    1     13
                  14    18    UNUSED
                  32    95    32
                  127   1     UNUSED
      BASESET     "ISO Registration Number 100//CHARSET
                  ECMA-94 Right Part of Latin Alphabet Nr. 1//ESC 2/13
      4/1"
      DESCSET     128   32    UNUSED
                  160   95    32
                  255   1     UNUSED

CAPACITY    SGMLREF
            TOTALCAP    150000
            GRPCAP              150000

SCOPE       DOCUMENT
SYNTAX
      SHUNCHAR CONTROLS       0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
18
                        19 20 21 22 23 24 25 26 27 28 29 30 31 127 255
      BASESET     "ISO 646:1983//CHARSET
            International Reference Version (IRV)//ESC 2/5 4/0"
      DESCSET     0     128    0
      FUNCTION    RE          13
                  RS          10
                  SPACE       32
                  TAB SEPCHAR  9
      NAMING            LCNMSTRT ""
                  UCNMSTRT ""
                  LCNMCHAR ".-"
                  UCNMCHAR ".-"
                  NAMECASE    GENERAL YES
                              ENTITY  NO
      DELIM       GENERAL     SGMLREF
                  SHORTREF    SGMLREF
      NAMES       SGMLREF
      QUANTITY    SGMLREF
                  NAMELEN     34
                  TAGLVL            100
                  LITLEN            1024
                  GRPGTCNT    150
                  GRPCNT            64

FEATURES
      MINIMIZE
            DATATAG  NO
            OMITTAG  NO
            RANK     NO
            SHORTTAG NO
      LINK
            SIMPLE   NO
```

```
            IMPLICIT NO
            EXPLICIT NO
      OTHER
            CONCUR      NO
            SUBDOC      NO
            FORMAL      YES
      APPINFO     NONE
>

<!DOCTYPE HTMLPLUS [

<!-- DTD for HTML+

Markup minimisation should be avoided, otherwise the default
<!SGML> declaration is fine.

Browsers should be forgiving of markup errors.

Common Attributes:

   id       the id attribute allows authors to name elements such as
            headers and paragraphs as potential destinations for links.
            Note that links don't specify points, but rather extended
            objects.

   index    allows authors to specify how given headers etc should be
            indexed as primary or secondary keys, where "/" separates
            primary from secondary keys, ";" separates multiple entries

-->

<!-- ENTITY DECLARATIONS

  <!ENTITY % foo "X | Y | Z"> is a macro definition for parameters and in
  subsequent statements, the string "%foo;" is expanded to "X | Y | Z"

  Various classes of SGML text types:

    #CDATA   text which doesn't include markup or entity references

    #RCDATA  text with entity references but no markup

    #PCDATA  text occurring in a context in which markup and entity
             references may occur.
-->

<!ENTITY % URL "CDATA" -- a URL or URN designating a hypertext node -->

<!ENTITY % text "#PCDATA|A|IMG|EM|EMBED|INPUT|BR|CHANGED|NOTE">

<!ENTITY % paras "P|PRE|FIG">

<!ENTITY % lists "UL|OL|DL">

<!ENTITY % misc "HR|TBL|FORM|PANEL|GROUP">

<!ENTITY % heading "H1|H2|H3|H4|H5|H6">

<!ENTITY % table "%text;|P|%heading;|%lists;">

<!ENTITY % main "%heading;|%misc;|%lists;|%paras;|%text;">

<!ENTITY % setup "(TITLE? & HTML? &ISINDEX? & NEXTID? & LINK* & BASE?)">

<!--

<!ELEMENT tagname - - CONTENT> elements needing closing tags

<!ELEMENT tagname - O CONTENT> elements without closing tags

<!ELEMENT tagname - O EMPTY> elements without content or closing tags

   The content definition is:

      a)    an entity definition as defined above

      b)    a tagname
```

```
      c)    (brackets enclosing the above)
   These may be combined with the operators:
      A*    A occurs zero or more times
      A+    A occurs one or more times
      A|B   implies either A or B
      A?    A occurs zero or one times
      A,B   implies first A then B
-->
<!ELEMENT HTMLPLUS O O ((HEAD, BODY) | ((%setup;), (%main;)*))>
<!ELEMENT HEAD - - (%setup;)>
<!ELEMENT BODY - - (%main;)*>
<!-- Document title -->
<!ELEMENT TITLE - - (#PCDATA | EM)+>
<!ATTLIST TITLE
      id       ID      #IMPLIED -- link destination --
      index    CDATA   #IMPLIED -- entries for index compilation -->
<!-- Document role for cataloging documents held by servers -->
<!ELEMENT HTML - O (EMPTY)>
<!ATTLIST HTML role CDATA #IMPLIED -- home page, index, ... -->
<!-- Floating panel which can be moved around relative to the normal text
flow. Often rendered with a different background and possibly framed. The
panel can be anchored to a named point in the document as specified by
the AT attribute. The panel may be placed at that point or after, but not
before.
-->
<!ELEMENT PANEL - - (TITLE?, (%main;)*)>
<!ATTLIST PANEL
      id       ID      #IMPLIED -- defines link destination --
      at       IDREF   #IMPLIED -- anchor point --
      role     CDATA   #IMPLIED -- annotation/aside
      index    CDATA   #IMPLIED -- entries for index compilation -->
<!ELEMENT HR - O EMPTY -- Horizontal Rule -->
<!-- Document headers -->
<!ELEMENT (%heading;) - - (#PCDATA | EM)+>
<!ATTLIST (%heading;)
      id       ID      #IMPLIED -- defines link destination --
      index    CDATA   #IMPLIED -- entries for index compilation -->
<!-- logical emphasis with optional style hints -->
<!ELEMENT EM - - (%text;)*>
<!ATTLIST EM
   role   CDATA  #IMPLIED -- semantic category e.g. CITE --
   b      (b)    #IMPLIED -- render in bold font --
   i      (i)    #IMPLIED -- render in italic font --
   u      (u)    #IMPLIED -- underline text --
   tt     (tt)   #IMPLIED -- render in typewriter font --
   tr     (tr)   #IMPLIED -- render in serif (Times Roman) font --
   hv     (hv)   #IMPLIED -- render in sans serif (Helvetica) font --
   sup    (sup)  #IMPLIED -- superscript --
   sub    (sub)  #IMPLIED -- subscript --
   index  CDATA  #IMPLIED -- entries for index compilation -->
```

```
<!-- Paragraphs with different roles and optional style hints
     Note that paragraphs act as containers for the following text -->

<!ELEMENT P - O (%text;)+>

<!ATTLIST P
   id       ID          #IMPLIED -- link destination --
   role     CDATA       #IMPLIED -- semantic role --
   align    CDATA       #IMPLIED -- left, center or right --
   indent   (indent)    #IMPLIED -- indented margins --
   index    CDATA       #IMPLIED -- entries for index compilation -->

<!ELEMENT BR - O EMPTY -- line break in normal text-->

<!-- Preformatted text with fixed pitch font, respecting original spacing
and newlines. Authors can also request proportional fonts. Further
control is possible with EM, and TAB -->

<!ELEMENT PRE - - (TAB|%text;)+>

<!ATTLIST PRE
   id       ID       #IMPLIED -- link destination --
   role     CDATA    #IMPLIED -- various styles --
   tr       (tr)     #IMPLIED -- serif (Times Roman) font --
   hv       (hv)     #IMPLIED -- sans serif (Helvetica) font --
   width    NUMBER   #IMPLIED -- e.g. 40, 80, 132 --
   index    CDATA    #IMPLIED -- entries for index compilation -->

<!ELEMENT TAB - O EMPTY>

<!ATTLIST TAB
   at  NUMBER  #IMPLIED -- position measured in widths of a capital M --
   align  (left|center|right|decimal)  left -- tab alignment -->

<!-- Lists which can be nested -->

<!ELEMENT OL - - (LI | UL | OL)+ -- ordered list -->

<!ATTLIST OL
   id       ID         #IMPLIED
   compact  (compact)  #IMPLIED
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!ELEMENT UL - - (LI | UL | OL)+ -- unordered list -->

<!ATTLIST UL
   id       ID         #IMPLIED      -- link destination --
   compact  (compact)  #IMPLIED      -- reduced interitem spacing --
   narrow   (narrow)   #IMPLIED      -- narrow perhaps multi columns --
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!-- List items for UL and OL lists -->

<!ELEMENT LI - O (P|%text;)+>

<!ATTLIST LI
   id       ID         #IMPLIED
   src      %URL;      #IMPLIED -- icon for use in place of bullet --
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!-- Definition Lists (terms + definitions) -->

<!ELEMENT DL - - (DT,DD)+ -- DT and DD *MUST* be paired -- > <!ATTLIST DL
   id       ID         #IMPLIED
   compact  (compact)  #IMPLIED
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!ELEMENT DT - O (%text;)+ -- term text -- >

<!ELEMENT DD - O (P|UL|OL|%text;)+ -- definition text -- >

<!ATTLIST (DT|DD)
   id        ID         #IMPLIED
   index     CDATA      #IMPLIED -- entries for index compilation -->
```

```
<!-- Tables with titles and column headers, e.g.

   <tbl border>
      <tt> An Example of a Table
      <th> <th s="2"> average <th> other <tr>
      <th> <th> height <th> weight <th> category <tr>
      <td> males <td> 1.9 <td> .003 <td> yyy <tr>
      <td> females <td> 1.7 <td> .002 <td> xxx
   </tbl>

-->

<!ELEMENT TBL - - (TT?, (TH|TD|TR|TB)*) -- mixed headers and data -->

<!ATTLIST TBL
   id       ID         #IMPLIED
   compact  (compact)  #IMPLIED      -- if present use compact style --
   border   (border)   #IMPLIED      -- if present draw borders --
   index    CDATA      #IMPLIED       -- entries for index compilation -->

<!ELEMENT TT - O (%text;)+ -- table title -->

<!ATTLIST TT top (top) #IMPLIED -- place title above table -->

<!ELEMENT TH - O (%table;)* -- a header cell -->

<!ATTLIST TH
   colspan  NUMBER     1          -- columns spanned --
   rowspan  NUMBER     1          --. rows spanned --
   align    CDATA    #IMPLIED     -- left, center or right -->

<!ELEMENT TD - O (%table;)* -- a data cell -->

<!ATTLIST TD
   colspan  NUMBER     1          -- columns spanned --
   rowspan  NUMBER     1          --. rows spanned --
   align    CDATA    #IMPLIED     -- left, center or right -->

<!ELEMENT TR - O EMPTY -- row separator -->

<!ELEMENT TB - O EMPTY -- vertical break of 1/2 line spacing -->

<!-- Forms composed from input fields and selection menus

These elements define fields which users can type into or select with
mouse clicks. The browser should manage the input focus e.g. with the
tab/shift tab keys and mouse clicks.

The enter/return key is then taken to mean the use has filled in the form
and wants the apppropriate action taken:
   - send as query/update to WWW server
   - email/fax to designated person

The action is specified as a URL, e.g. "mailto:dsr@hplb.hpl.hp.com You
can specify additional mail headers with the MH tag:

<MH>Subject: Please add me to tennis tournament</MH>

Each FORM should include one or more INPUT elements which can be layed
out with normal and preformatted text, lists, and tables.
 -->

<!ELEMENT FORM - - (MH, (%main;)*)>

<!ATTLIST FORM
   id       ID         #IMPLIED
   action   %URL;      #IMPLIED
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!ELEMENT MH - - CDATA -- one or more RFC 822 header fields -->

<!ATTLIST MH hidden (hidden) #IMPLIED -- hide the mail headers from view -->
```

```
<!-- INPUT elements should be defined within a FORM element.

Users can alter the value of the INPUT element by typing or clicking with
the mouse. Use radio buttons for selecting one attribute value from a set
of alternatives. In this case there will be several INPUT elements with
the same name. Attributes which can take multiple values at the same time
should be defined with checkboxes: define each allowed value in a
separate INPUT element but with the same attribute name. For checkboxes
and radio buttons, the value doesn't change, instead the state of the
button shown by the presence or absence of the checked attribute in each
element.

The size attribute specifies the size of the input field as appropriate
to each type. For text this gives the width in characters and height in
lines (separated by an "x"). For numbers this gives the maximum
precision.

-->

<!ELEMENT INPUT - O EMPTY>

<!ATTLIST INPUT
   name     CDATA     #IMPLIED -- attribute name (may not be unique) --
   type     CDATA     #IMPLIED --TEXT,URL,INT,FLOAT,DATE,CHECKBOX,RADIO--
   size     CDATA     #IMPLIED -- e.g."32x4" for multiline text --
   value    CDATA     #IMPLIED -- attribute value (altered by user) --
   checked  (checked) #IMPLIED -- for check boxes and radio buttons --
   disabled(disabled)   #IMPLIED    -- if grayed out --
   error    (error)   #IMPLIED -- if in error -->

<!-- Embedded Data

You can embed information in a foreign format into the HTML+ document.
This is very convenient for mathematical equations and simple drawings.
Images and complex drawings are better specified as linked documents
using the FIG or IMG elements.

Arbitrary 8 bit data is allowed but any occurrences of the following
chars must be escaped as shown:

      "&"      by      "&amp;"
      "<"      by      "&lt;"
      ">"      by      "&gt;"
The browser can pipe such data thru filters to generate the corresponding
pixmap The data format is specified as a MIME content type, e.g.
"text/eqn"


It would have been nice to use the NOTATION type in place of CDATA, but
this doesn't
appear to be practical with externally specified content type names.
-->

<!ELEMENT EMBED - - (RCDATA)>
<!ATTLIST EMBED
   id       ID        #IMPLIED
   type     CDATA     #IMPLIED  -- mime content type --
   index    CDATA     #IMPLIED -- entries for index compilation -->

<!-- Figures

The image/drawing is specified by a URL or as embedded data for simple
drawings. The element's text serves as the caption. Use the emphasis with
style = "credits" to record photo credits etc.

-->
```

```
<!ELEMENT FIG - - (EMBED?, FIGD?, (FIGA|FIGT)*, (%text;)*)>

<!ATTLIST FIG
   id       ID         #IMPLIED
   align    CDATA      #IMPLIED -- position: left, right or center --
   cap      CDATA      #IMPLIED -- caption at left, right, top, bottom --
   noflow   (noflow)   #IMPLIED -- disables text flow --
   ismap    (ismap)    #IMPLIED -- server can handle mouse clicks/drags --
   src      %URL;      #IMPLIED -- link to image data --
   index    CDATA      #IMPLIED -- entries for index compilation -->

<!ELEMENT FIGD - - (%table;) -- figure description -->

<!-- Figure anchors designate polygonal areas on the figure which can be
clicked with the mouse. The default area is the whole of the figure. This
mechanism interprets mouse clicks locally, and browsers can choose to
highlight the designated area (or change the mouse sprite) when the mouse
is moved over the area.

Note that polygons may be non-convex or even intersect themselves,
thereby complicating the definition of what is enclosed by the polygon.
Holes are excluded.
-->

<!ELEMENT FIGA - O EMPTY>

<!ATTLIST FIGA
   href     %URL;     #REQUIRED -- link to traverse when clicked --
   area     NUMBERS   #IMPLIED -- x1,y1,x2,y2,x3,y3,... -->

<!-- FIGT Text on top of an figure background, or in a colored background
box which sits arbitrarily on top of an figure background. The text can
include headers, lists and tables etc. The width attribute allows you to
limit the width of the text box. The height is then determined
automatically by the browser.

FIGT can also be used to position a graphic on top of a picture using an
IMG element within FIGT. In this case the chromakey attribute may allow
parts of the underlying image to show through.

You can make the whole of the box into a hypertext link. This will act as
if it is underneath any hypertext links specified by the overlay markup
itself.

FIGT can also be used for annotations on figures, and held separately
from the document with the figure to which they apply. In this case, use
the idref attribute to name which figure is appropriate.
-->

<!ELEMENT FIGT - - (%main;)>

<!ATTLIST FIGT
   idref  IDREF    #IMPLIED -- names FIG element if held separately --
   at     NUMBERS  #IMPLIED -- upper left origin for text --
   width  NUMBER   #IMPLIED -- given as fraction of picture --
   framed (framed) #IMPLIED -- framed with coloured background --
   href   %URL;    #IMPLIED -- link to traverse when clicked -->

<!-- inline icons/small graphics
The align attribute defines whether the top middle or bottom of the
graphic and current text line should be aligned vertically

The SEETHRU attribute is intended as a chromakey to allow a given colour
to be designated as "transparent". Pixels with this value should not be
painted. The exact format of this attribute's value has yet to be
defined.


Use the FIG tag for captioned figures with active areas etc.
-->

<!ELEMENT IMG - O EMPTY>
```

```
<!ATTLIST IMG
   src     %URL;   #REQUIRED -- where to get image data --
   align   CDATA   #IMPLIED  -- top, middle or bottom --
   seethru CDATA   #IMPLIED  -- for transparency --
   ismap   (ismap) #IMPLIED  -- send mouse clicks/drags to server -->

<!-- Hierarchical groups for books, chapters, sections etc. -->

<!ELEMENT GROUP - - ((TITLE|LINK*), (%main;)*)>

<!ATTLIST GROUP
   id    ID       #IMPLIED
   role  CDATA    #IMPLIED -- book, chapter, section etc. --
   inset (inset)  #IMPLIED -- rendering hint: indent margins -->

<!-- change bars defined by a matched pair of CHANGED elements:

      <changed id=z34> changed text <changed idref=z34>

This tag can't act as a container, since changes don't respect
the nesting implied by paragraphs, headers, lists etc.
-->

<!ELEMENT CHANGED - O EMPTY>

<!ATTLIST CHANGED -- one of id and idref is always required --
   id       ID     #IMPLIED -- signals start of changes --
   idref    IDREF  #IMPLIED -- signals end of changes -->

<!-- Matched pairs of NOTE elements can be used to delimit text
associated with one or more annotations. The annotation itself can be
linked via an explicit URL or defined as one or more PANEL elements
naming this NOTE.

      <note id=z34> delimited text <note idref=z34>

This tag can't act as a container, since users may select text without
regard to
the nesting implied by paragraphs, headers, lists etc.
-->

<!ELEMENT NOTE - O EMPTY>

<!ATTLIST NOTE -- one of id and idref is always required --
   id       ID     #IMPLIED -- signals start of delimited text --
   href     %URL;  #IMPLIED -- for directly specifying the annotation --
   idref    IDREF  #IMPLIED -- signals end of delimited text -->

<!-- Hypertext Links from points within document nodes -->

<!ELEMENT A - - (#PCDATA | IMG | EM | EMBED)*>

<!ATTLIST A
   id      ID       #IMPLIED -- as target of link --
   name    CDATA    #IMPLIED -- for backwards compatibility with HTML--
   href    %URL;    #IMPLIED -- destination node --
   role    CDATA    #IMPLIED -- role of link, e.g. "partof" --
   effect  CDATA    #IMPLIED -- replace/new/overlay --
   print   CDATA    #IMPLIED -- reference/footnote/section --
   title   CDATA    #IMPLIED -- when otherwise unavailable --
   type    CDATA    #IMPLIED -- for presentation cues --
   size    NAMES    #IMPLIED -- for progress cues -->
```

```
<!-- Other kinds of relationships between documents -->

<!ELEMENT LINK - O EMPTY>

<!ATTLIST LINK
   href     %URL;     #IMPLIED -- destination node --
   role     CDATA     #IMPLIED -- role played, e.g. "toc" -->

<!-- Original document URL for resolving relative URLs  -->

<!ELEMENT BASE - O EMPTY>
<!ATTLIST BASE HREF %URL; #IMPLIED>

<!-- Signifies the document's URL accepts queries -->

<!ELEMENT ISINDEX - O (EMPTY)>
<!ATTLIST ISINDEX href %URL; #IMPLIED -- defaults to document's URL -->

<!-- For use with autonumbering editors - don't reuse ids, allocate next
one starting from this one -->

<!ELEMENT NEXTID - O (EMPTY)>
<!ATTLIST NEXTID N NAME #REQUIRED>

<!-- Mnemonic character entities for 8 bit ANSI Latin-1 -->

<!ENTITY iexcl "&#161;" -- inverted exclamation mark -->
<!ENTITY cent "&#161;" -- cent sign -->
<!ENTITY pound "&#163;" -- pound sign -->
<!ENTITY yen "&#165;" -- yen sign -->
<!ENTITY brvbar "&#166;" -- broken vertical bar -->
<!ENTITY sect "&#167;" -- section sign -->
<!ENTITY copy "&#169;" -- copyright sign -->
<!ENTITY laquo "&#171;" -- angle quotation mark, left -->
<!ENTITY raquo "&#187;" -- angle quotation mark, right -->
<!ENTITY not "&#172;" -- negation sign -->
<!ENTITY reg "&#174;" -- circled R registered sign -->
<!ENTITY deg "&#176;" -- degree sign -->
<!ENTITY plusmn "&#177;" -- plus or minus sign -->
<!ENTITY sup2 "&#178;" -- superscript 2 -->
<!ENTITY sup3 "&#179;" -- superscript 3 -->
<!ENTITY micro "&#181;" -- micro sign -->
<!ENTITY para "&#182;" -- paragraph sign -->
<!ENTITY sup1 "&#185;" -- superscript 1 -->
<!ENTITY middot "&#183;" -- center dot -->
<!ENTITY frac14 "&#188;" -- fraction 1/4 -->
<!ENTITY frac12 "&#189;" -- fraction 1/2 -->
<!ENTITY iquest "&#191;" -- inverted question mark -->
<!ENTITY frac34 "&#190;" -- fraction 3/4 -->
<!ENTITY AElig "&#198;" -- capital AE diphthong (ligature) -->
<!ENTITY Aacute "&#193;" -- capital A, acute accent -->
<!ENTITY Acirc "&#194;" -- capital A, circumflex accent -->
<!ENTITY Agrave "&#192;" -- capital A, grave accent -->
<!ENTITY Aring "&#197;" -- capital A, ring -->
<!ENTITY Atilde "&#195;" -- capital A, tilde -->
<!ENTITY Auml "&#196;" -- capital A, dieresis or umlaut mark -->
<!ENTITY Ccedil "&#199;" -- capital C, cedilla -->
<!ENTITY ETH "&#208;" -- capital Eth, Icelandic -->
<!ENTITY Eacute "&#201;" -- capital E, acute accent -->
<!ENTITY Ecirc "&#202;" -- capital E, circumflex accent -->
<!ENTITY Egrave "&#200;" -- capital E, grave accent -->
<!ENTITY Euml "&#203;" -- capital E, dieresis or umlaut mark -->
<!ENTITY Iacute "&#205;" -- capital I, acute accent -->
<!ENTITY Icirc "&#206;" -- capital I, circumflex accent -->
<!ENTITY Igrave "&#204;" -- capital I, grave accent -->
<!ENTITY Iuml "&#207;" -- capital I, dieresis or umlaut mark -->
<!ENTITY Ntilde "&#209;" -- capital N, tilde -->
<!ENTITY Oacute "&#211;" -- capital O, acute accent -->
<!ENTITY Ocirc "&#212;" -- capital O, circumflex accent -->
<!ENTITY Ograve "&#210;" -- capital O, grave accent -->
```

O I P E JC64
OCT 24 2003
PATENT & TRADEMARK OFFICE

```
<!ENTITY Oslash "&#216;" -- capital O, slash -->
<!ENTITY Otilde "&#213;" -- capital O, tilde -->
<!ENTITY Ouml "&#214;" -- capital O, dieresis or umlaut mark -->
<!ENTITY THORN "&#222;" -- capital THORN, Icelandic -->
<!ENTITY Uacute "&#218;" -- capital U, acute accent -->
<!ENTITY Ucirc "&#219;" -- capital U, circumflex accent -->
<!ENTITY Ugrave "&#217;" -- capital U, grave accent -->
<!ENTITY Uuml "&#220;" -- capital U, dieresis or umlaut mark -->
<!ENTITY Yacute "&#221;" -- capital Y, acute accent -->
<!ENTITY aacute "&#225;" -- small a, acute accent -->
<!ENTITY acirc "&#226;" -- small a, circumflex accent -->
<!ENTITY aelig "&#230;" -- small ae diphthong (ligature) -->
<!ENTITY agrave "&#224;" -- small a, grave accent -->
<!ENTITY amp "&amp;" -- ampersand -->
<!ENTITY aring "&#229;" -- small a, ring -->
<!ENTITY atilde "&#227;" -- small a, tilde -->
<!ENTITY auml "&#228;" -- small a, dieresis or umlaut mark -->
<!ENTITY ccedil "&#231;" -- small c, cedilla -->
<!ENTITY eacute "&#233;" -- small e, acute accent -->
<!ENTITY ecirc "&#234;" -- small e, circumflex accent -->
<!ENTITY egrave "&#232;" -- small e, grave accent -->
<!ENTITY eth "&#240;" -- small eth, Icelandic -->
<!ENTITY euml "&#235;" -- small e, dieresis or umlaut mark -->
<!ENTITY gt "&#62;" -- greater than -->
<!ENTITY iacute "&#237;" -- small i, acute accent -->
<!ENTITY icirc "&#238;" -- small i, circumflex accent -->
<!ENTITY igrave "&#236;" -- small i, grave accent -->
<!ENTITY iuml "&#239;" -- small i, dieresis or umlaut mark -->
<!ENTITY lt "&lt;" -- less than -->
<!ENTITY ntilde "&#241;" -- small n, tilde -->
<!ENTITY oacute "&#243;" -- small o, acute accent -->
<!ENTITY ocirc "&#244;" -- small o, circumflex accent -->
<!ENTITY ograve "&#242;" -- small o, grave accent -->
<!ENTITY oslash "&#248;" -- small o, slash -->
<!ENTITY otilde "&#245;" -- small o, tilde -->
<!ENTITY ouml "&#246;" -- small o, dieresis or umlaut mark -->
<!ENTITY szlig "&#223;" -- small sharp s, German (sz ligature) -->
<!ENTITY thorn "&#254;" -- small thorn, Icelandic -->
<!ENTITY uacute "&#250;" -- small u, acute accent -->
<!ENTITY ucirc "&#251;" -- small u, circumflex accent -->
<!ENTITY ugrave "&#249;" -- small u, grave accent -->
<!ENTITY uuml "&#252;" -- small u, dieresis or umlaut mark -->
<!ENTITY yacute "&#253;" -- small y, acute accent -->
<!ENTITY yuml "&#255;" -- small y, dieresis or umlaut mark -->


<!-- other entities -->

<!ENTITY ndash "--"  -- En dash -->
<!ENTITY mdash "---" -- Em dash -->

<!ENTITY nbsb " " -- non breaking space -->
<!ENTITY ensp "  " -- En space -->
<!ENTITY emsp "    " -- Em space -->

<!ENTITY shy "-" -- soft hyphen -->


<!-- The END -->

]>
```

OIPE JC64
OCT 24 2003
PATENT & TRADEMARK OFFICE

Appendix II - Entity Definitions

The following character definitions conform to widely available 8 bit character sets, e.g. the ANSI Latin-1 character set which is available for both the PC and X11. The corresponding 8-bit character codes are given in the DTD. All definitions conform to the ISO 8879-1986 naming conventions.

Æ	capital AE diphthong (ligature)
Á	capital A, acute accent
Â	capital A, circumflex accent
À	capital A, grave accent
Å	capital A, ring
Ã	capital A, tilde
Ä	capital A, dieresis or umlaut mark
Ç	capital C, cedilla
Ð	capital Eth, Icelandic
É	capital E, acute accent
Ê	capital E, circumflex accent
È	capital E, grave accent
Ë	capital E, dieresis or umlaut mark
Í	capital I, acute accent
Î	capital I, circumflex accent
Ì	capital I, grave accent
Ï	capital I, dieresis or umlaut mark
Ñ	capital N, tilde
Ó	capital O, acute accent
Ô	capital O, circumflex accent
Ò	capital O, grave accent
Ø	capital O, slash
Õ	capital O, tilde
Ö	capital O, dieresis or umlaut mark
Þ	capital THORN, Icelandic
Ú	capital U, acute accent
Û	capital U, circumflex accent
Ù	capital U, grave accent
Ü	capital U, dieresis or umlaut mark
Ý	capital Y, acute accent
á	small a, acute accent
â	small a, circumflex accent
æ	small ae diphthong (ligature)
à	small a, grave accent
å	small a, ring
ã	small a, tilde
ä	small a, dieresis or umlaut mark
ç	small c, cedilla
é	small e, acute accent
ê	small e, circumflex accent
è	small e, grave accent
ð	small eth, Icelandic

ë	small e, dieresis or umlaut mark
í	small i, acute accent
î	small i, circumflex accent
ì	small i, grave accent
ï	small i, dieresis or umlaut mark
ñ	small n, tilde
ó	small o, acute accent
ô	small o, circumflex accent
ò	small o, grave accent
ø	small o, slash
õ	small o, tilde
ö	small o, dieresis or umlaut mark
ß	small sharp s, German (sz ligature)
þ	small thorn, Icelandic
ú	small u, acute accent
û	small u, circumflex accent
ù	small u, grave accent
ü	small u, dieresis or umlaut mark
ý	small y, acute accent
ÿ	small y, dieresis or umlaut mark

In addition, there are some common publishing characters:

¡	inverted exclamation mark
¢	cent sign
£	pound sign
¥	yen sign
¦	broken vertical bar
§	section sign
©	copyright sign
«	angle quotation mark, left
»	angle quotation mark, right
¬	negation sign
®	circled R registered sign
°	degree sign
±	plus or minus sign
²	superscript 2
³	superscript 3
µ	micro sign
¶	paragraph sign
¹	superscript 1
·	center dot
¼	fraction 1/4
½	fraction 1/2
¾	fraction 3/4
¿	inverted question mark

Finally, there are several special purpose definitions:

–	En sized horizontal dash (--)
—	Em sized horizontal dash (---)
	Non-breaking space
	En space ( )
	Em space ( )
	Soft hyphen

C I P E JC64
OCT 24 2003
PATENT & TRADEMARK OFFICE

Appendix III - Compatibility with HTML

HTML+ browsers should be able to view HTML documents with very little extra code and it is strongly recommended that browsers support both formats. Older HTML browsers will be able to view HTML+ documents which don't contain figures, tables or forms.

Lists

HTML	HTML+
<menu>	<ul compact>
<dir>	<ul narrow>

Emphasis

HTML+ replaces the various tags used by HTML with a single tag. It may be worth changing the name for the emphasis tag in HTML+ from EM to EM, to gain compatibility with this common form. However, using EM might be confused with the typographical term *em* as in em dash (you also get en dash). EM has the merit of being unambigous.

HTML	HTML+
<tt>	<em tt>
	<em b>
<b>	<em b>
<i>	<em i>
<u>	<em u>
<code>	
<samp>	
<kbd>	
<var>	
<dfn>	
<cite>	

Miscellaneous

Some tags which are deprecated in HTML are now obsolete, and should be mapped to preformatted text. This doesn't work quite right as PRE assumes that characters such as "<", ">" and "&" have been replaced by their entity definitions. Browsers should perhaps treat "<" as verbatim unless it forms part of an expected tag. In this way, unescaped occurrences of these three characters will normally display as intended.

HTML	HTML+
<plaintext>	<pre>
<xmp>	<pre>
<listing>	<pre>

The following two tags have been absorbed into the standard mechanism for paragraphs:

<address>	becomes	<p role="byline" align="right">
<blockquote>	becomes	<p role="quote">

Notes for Implementors

Please ensure that browsers can tolerate bad markup. In practice, this is quite straightforward to achieve, provided a naive top-down SGML parser is avoided. A forgiving parser should be able to cope with tags in unexpected positions, e.g. the <A> tag bracketing a header[16]. Unknown tags should be simply ignored.

Implementors should endevour to make sure that documents can be scrolled efficiently regardless of their length. Always parsing from the start of the document leads to jerky performance. Two strategies for efficiently scrolling through documents are:

a) Establish regular landmarks throughout the document for which the state of the parse is known. The browser can then work forward from the nearest landmark, when it needs to refresh the screen after a scroll operation. The landmarks need updating when users make changes, while using a WYSIWYG editor.

b) When scrolling up, parse backwards to work out the state at earlier points in the document. This can be done via a combination of skipping back, looking for markup which causes a line break etc. and then parsing forward until the current position, to find the change of state. This can be repeated until the parser reaches a point prior to the new top of the window.

Practical experience has shown the importance of providing cues to users on progress in retrieving documents over the network. These will depend on the protocol, but should show at least how much data has been received at any point. The network connections shouldn't block, and an abort button is essential[17]. It is generally better to avoid displaying the retrieved document in a new window, unless explicitly requested by the user, e.g. by holding down the shift key when clicking the hypertext link.

References

This is missing the appropriate references to work on the syntax and name service for URNs. The HTTP definition needs updating to cover the encoding of form data (and *ismap* ?).

[Berners-Lee 93a] "*Hypertext Markup Language (HTML)*", Tim Berners-Lee, March 1993.
URL=`ftp://info.cern.ch/pub/www/doc/http-spec.ps`

[Berners-Lee 93b] "*Uniform Resource Locators*", Tim Berners-Lee, January 1992.
URL=`ftp://info.cern.ch/pub/ietf/url4.ps`

[Berners-Lee 93c] "*Protocol for the Retrieval and Manipulation of Textual and Hypermedia Information*", Tim Berners-Lee, 1993.
URL=`ftp://info.cern.ch/pub/www/doc/html-spec.ps`

[Goldfarb 90] "*The SGML Handbook*", Charles F. Goldfarb, Clarendon Press · Oxford.

[Kimber 93] Article in comp.text.sgml newsgroup, 24th May 1993 by Elliot Kimber (drmacro@vnet.almaden.ibm.com),
URL=`news:19930524.152345.29@almaden.ibm.com`

[Raisch 93] "*Style sheets for HTML*", Robert Raisch, June 1993, O'Reilly & Associates
email: `raisch.ora.com`

---

[16]Headers typically cause a line break and leave a vertical gap. If the hypertext link definition is parsed prior to the beginning of the header, the starting position for the button will be in the wrong place - browsers should therefore adjust this position to the beginning of the text.

[17]For X11 on Unix systems, the *select* system call can be used with non-blocking I/O to poll the event queue at regular intervals. The *XtAddInput* call acts as a wrapper around *select* for this very purpose. Users can then continue to view the current document as well as being able to click an abort button (which sets a global variable, polled by the comms software). Be careful to disable unsafe actions, e.g. trying to get a second document while still waiting to get the first (a race hazard).

OCT 24 2003 PATENT & TRADEMARK OFFICE

HTML+ support for eqn & Postscript

Dave_Raggett <dsr@hplb.hpl.hp.com>

- **Mail folder:** WWW Talk Apr-Jun 1993 Archives
- **Next message:** Marc Andreessen: "Re: SPaces and Tabs in HTML documents"
- **Previous message:** Tim Berners-Lee: "Re: SPaces and Tabs in HTML documents"
- **Reply:** Torben Noerup Nielsen: "re: HTML+ support for eqn & Postscript"
- **Reply:** Bill Janssen: "Re: HTML+ support for eqn & Postscript"
- **Reply:** Bill Janssen: "Re: HTML+ support for eqn & Postscript"


```
From: Dave_Raggett <dsr@hplb.hpl.hp.com>
Message-id: <9306140936.AA00563@manuel.hpl.hp.com>
Subject: HTML+ support for eqn & Postscript
To: torben@hawaii.edu, janssen@parc.xerox.com
Date: Mon, 14 Jun 93 10:36:40 BST
Cc: www-talk@nxoc01.cern.ch
Mailer: Elm [revision: 66.36.1.1]
```

Torben Nielsen says:

> I realize this has come up before, but how about really doing something about
> equation support? There are lots of documents I would like to put into the
> Web, but without support for embedded equations, it's really quite difficult.
> Something simple like eqn support would be great. And eqn shouldn't be all
> that hard to parse either.....

Bill Janssen chips in with:

> And I really like to send encapsulated Postcript in my documents...
> Having ghostscript should make the parsing and layout easy!

Well both of these will be possible with the HTML+ DTD, by using the capability to embed foreign formats inline in the HTML+ source, e.g.

```
      <H2>A example of an equation</H2>

      <EMBED TYPE="text/eqn">zeta (s) ~=~ sum from k=1 to inf
       k sup -s ~~~ (Re s &gt; 1) </EMBED>
```

The browser identifies the format of the embedded data from the "type" attribute, specified as a MIME content type. Certain characters need to be escaped using entity definitions, e.g. ">" by ">" in the example.

Building in support for a range of formats has the danger of leading to very large programs for browsers. This could be avoided by using a common API for rendering foreign formats, e.g. as functions that take a sequence of bytes and return a pixmap.

Browsers can then be upgraded to display new formats without changing their code at all. All you would need is a way of binding the MIME content type to the function name for that format, e.g. via X resources or a config file. The functions could be implemented as separate programs driven via pipes and stdin/stdout or as dynamically linked library modules (Windows DLLs).

How does that sound?

Dave

p.s. you can also put the foreign data in a separate file referenced by a URL.

Exhibit C

[redacted]

ALL-STATE LEGAL® 800-222-0510

OIPE
OCT 2 4 2003
PATENT & TRADEMARK OFFICE

WWW Workshop Announcement

Dave_Raggett <dsr@hplb.hpl.hp.com>

- **Mail folder:** WWW Talk Jul-Oct 1993
- **Next message:** Brian Smithson: "Re: Search keyword dialog box in Mosaic 2.0pre0?"
- **Previous message:** David Martland: "Mail messages in Mosaic"


```
From: Dave_Raggett <dsr@hplb.hpl.hp.com>
Message-id: <9307231606.AA26875@manuel.hpl.hp.com>
Subject: WWW Workshop Announcement
To: www-talk@nxoc01.cern.ch
Date: Fri, 23 Jul 93 17:06:18 BST
Mailer: Elm [revision: 66.36.1.1]
Status: RO

For those of you off to Cambridge for the workshop, there is a revised
version (dated 23rd July) of the HTML+ spec at:

        ftp://15.254.100.100/pub/htmlplus.ps

An older version can be found at:

        http://info.cern.ch/hypertext/WWW/MarkUp/htmlplus.ps

Can I suggest that you try and print off a copy to take with you.
Unfortunately I won't be able to come myself, but hope that there will
be discussion of the spec and look forward to finding out how you all
get on.

I am hoping to get the HTML+ spec accepted as an Internet Draft as soon as
practical, and would very much like help with checking the spec for technical
feasibility and improving clarity etc.

Have fun,

Dave Raggett

p.s. There seems to be a Ctrl-D at the start of the file, an annoying
artifact generated by Microsoft Word! This shouldn't cause problems though.
```

Exhibit D

RECEIVED
OCT 2 9 2003
OFFICE OF PETITIONS

PATENT & TRADEMARK OFFICE
OCT 24 2003

OIPE
OCT 24 2003
PATENT & TRADEMARK OFFICE

Re: Lets keep the web together

marca@ncsa.uiuc.edu (Marc Andreessen)

- **Mail folder:** WWW Talk 1992 Archives
- **Next message:** Robert Raisch: "Re: Lets keep the web together "
- **Previous message:** Dan Connolly: "Re: Lets keep the web together "
- **In-reply-to:** Dan Connolly: "Re: Lets keep the web together "
- **References:** Tim Berners-Lee: "Lets keep the web together"

Date: Tue, 1 Dec 92 12:15:42 -0800
From: marca@ncsa.uiuc.edu (Marc Andreessen)
Message-id: <9212012015.AA22660@wintermute.ncsa.uiuc.edu>
To: Dan Connolly <connolly@pixel.convex.com>
Cc: timbl@nxoc01.cern.ch, www-talk@nxoc01.cern.ch
Subject: Re: Lets keep the web together
In-reply-to: <9212011804.AA24775@pixel.convex.com>
References: <9212011552.AA01907@www3.cern.ch>
<9212011804.AA24775@pixel.convex.com>

Dan Connolly writes:
> You can't use any of the parts without using the whole thing: you
> can't use the parser unless you use the HText data structures,
> including the HTMainText global variable! [...]

I have to agree with this as a negative assessment. Trying to figure out how to deal with the library hasn't been very much fun.

> Again, we see the need for a CSCW platform.

Well, for what it's worth, the server running on hoohoo.ncsa.uiuc.edu:80 will accept the ANNOTATE command as proposed in a previous message of mine, and I can send the patches to enable this (as well as modify the linemode browser and library to provide annotation functionality) to anyone who wants them. A more complex system is certainly desirable, though.

> I think that if we archived the www-talk mailing list, and built an
> HTTP server that would serve the articles up with hypertext links
> between folloups etc, it would be a good start.
>
> If I code up a server in perl that can do this, will somebody
> provide disk space and network access?

I would, but it already exists:

http://eies2.njit.edu:80/wmail.html

Dan, if you want to do a mailing list archive also, contact me and we'll set it up...

Marc

RECEIVED
OCT 29 2003
OFFICE OF PETITIONS